Universidade Federal do Rio de Janeiro

Núcleo de Computação Eletrônica

Renata Gomes Barbosa de Andrade

O ESTADO DA ARTE DE SEGURANÇA ADAPTATIVA

Rio de Janeiro
2009

Renata Gomes Barbosa de Andrade

O ESTADO DA ARTE DE SEGURANÇA ADAPTATIVA

Monografia apresentada para obtenção do título de Especialista em Gerência de Redes de Computadores no Curso de Pós-Graduação Lato Sensu em Gerência de Redes de Computadores e Tecnologia Internet do Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro – NCE/UFRJ.

Orientador:
Luci Pirmez, D.Sc. – COPPE/UFRJ – Brasil - 1996

Rio de Janeiro

2009

Renata Gomes Barbosa de Andrade

O ESTADO DA ARTE DE SEGURANÇA ADAPTATIVA

Monografia apresentada para obtenção do título de Especialista em Gerência de Redes de Computadores no Curso de Pós-Graduação Lato Sensu em Gerência de Redes de Computadores e Tecnologia Internet do Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro – NCE/UFRJ.

Aprovada em outubro de 2009.

Luci Pirmez

Luci Pirmez, D.Sc., UFRJ, Brasil

AGRADECIMENTOS

Acima de tudo, agradeço a Deus, pela sua presença constante em minha vida.

À professora e orientadora Luci Pirmez, pelos ensinamentos e incentivos que tornaram possível a conclusão desta monografia.

À professora e amiga Maria Irene Sá, por todo seu apoio, incentivo e pela confiança depositada em mim.

Ao Núcleo de Computação Eletrônica, por me conceder a bolsa de estudos.

Aos meus pais, por todo amor, carinho e dedicação à minha criação.

Ao meu marido Rafael, pelo seu amor, companheirismo e compreensão nas horas difíceis.

À minha avó Zita, que me ensinou os primeiros passos no mundo do conhecimento.

À amiga Linair Campos, pela ajuda na revisão do texto e pelas sugestões.

À amiga Fabrícia Sobral, por todo seu apoio no meu dia a dia.

Enfim, a todos que de alguma maneira contribuíram para a execução desse trabalho, seja pela ajuda constante ou por uma palavra de amizade.

Muito obrigada!

RESUMO

ANDRADE, Renata Gomes Barbosa de. **O ESTADO DA ARTE DE SEGURANÇA ADAPTATIVA**. Monografia (Especialização em Gerência de Redes e Tecnologia Internet). Núcleo de Computação Eletrônica, Universidade Federal do Rio de Janeiro. Rio de Janeiro, 2009.

A necessidade da mobilidade, aliada ao aumento do uso das redes sem fio através dos dispositivos portáteis impõem a criação de um alicerce de sustentação dos requisitos de segurança adaptativo. Este trabalho apresenta o Estado da Arte de Segurança Adaptativa, definindo políticas eficientes para alcançar um elevado nível de segurança das aplicações e das infra-estruturas de Tecnologia da Informação.

ABSTRACT

ANDRADE, Renata Gomes Barbosa de. **O ESTADO DA ARTE DE SEGURANÇA ADAPTATIVA**. Monografia (Especialização em Gerência de Redes e Tecnologia Internet). Núcleo de Computação Eletrônica, Universidade Federal do Rio de Janeiro. Rio de Janeiro, 2009.

The necessity of mobility, and the increase of wireless networks use through mobile devices, imposes the creation of a foundation to support adaptive security requirements. This paper shows the State of the Art of Adaptive Security, defining efficient policies to achieve a high level application and Information Technology infrastructure security.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
ARS	Awaneress and Reactive Service
ASI	Adaptive Security Infrastructure
ATNAC	Adaptive Trust Negotiation Framework
CMS	Context Management Service
DOS	Deny of Service
ECA	Event-Condiction-Action
IDS	Intrusion Detection System
MAS	Mobile Application Servers
PDA	Personal Digital Assistants
PHA	Pervasive Hierarchy Assumption
SDT	Software Dynamic Translation
SLA	Service Level Agreement
TI	Tecnologia da Informação
TPM	Trusted Platform Module

SUMÁRIO

1 INTRODUÇÃO

A recente proliferação de dispositivos portáteis (celulares, laptops, PDAs, etc.) e a popularização das tecnologias para comunicação móvel levaram ao surgimento de aplicações e ambientes de computação ubíquos [5].

Essas aplicações devem considerar as mudanças dos requisitos dos usuários, da aplicação e da própria rede em seu estado de execução, para contornar ou diminuir o impacto das alterações nas operações. Tais alterações fazem com que as aplicações ubíquas necessitem mudar seus comportamentos a fim de se adaptarem às novas condições de execução. Para isso, elas precisam ser cientes de seus contextos de execução. Uma aplicação é ciente de contexto se usar o contexto para prover informações ou serviços ao usuário [17]. Segundo Springer em [23], contexto é qualquer tipo de informação acerca do ambiente de execução de uma aplicação que possa ser usada para aprimorar seu comportamento, fazendo-a operar de maneira adaptativa, personalizada, autônoma e flexível. Como não é desejável que o funcionamento de aplicações ubíquas seja interrompido para realizar adaptações nas próprias aplicações, é necessário que as adaptações sejam feitas de forma dinâmica e transparente. Dessa necessidade, surge o conceito de adaptação dinâmica, que consiste em alterar o comportamento da aplicação, ou seja, alterar o código que implementa as funcionalidades providas para os usuários, durante sua execução de forma transparente ([25], [26], e [27]).

Na presença de um ambiente altamente dinâmico e heterogêneo, surge um desafio relacionado à segurança destas aplicações, pois os controles de segurança variam conforme o contexto e é necessário gerenciar os requisitos de integridade, disponibilidade e confidencialidade da aplicação. Observa-se ainda que na literatura a importância de prover requisitos de segurança adaptativos não é exatamente uma novidade ([32]; [41]). Entretanto, a necessidade de maior mobilidade, aliada ao aumento do uso das redes sem fio através de dispositivos portáteis impõem a criação de um novo patamar no que diz respeito ao tratamento da segurança. Uma mudança no ambiente de execução que acarrete um aumento do risco de incidentes de segurança (ex: aumento de vulnerabilidades), é razão suficiente para implicar um acréscimo de controles de segurança no próprio dispositivo (comportamento pró-ativo) de forma a garantir um ambiente seguro para as aplicações ubíquas.

Outro desafio relacionado à execução de aplicações ubíquas consiste em tratar eventos que não foram considerados durante o projeto de um serviço de segurança. Não é possível, em tempo de projeto, antecipar todas as respostas que um serviço de segurança ciente de contexto pode dar frente à ocorrência de eventos ainda não previstos, uma vez que novas

vulnerabilidades são descobertas todos os dias. Para superar esses desafios, todo o alicerce de sustentação dos requisitos de segurança das aplicações deve ser adaptado segundo o ambiente de execução, levando-se em conta propriedades dinâmicas e heterogêneas que reflitam os diferentes ambientes computacionais envolvidos. Isso envolve uma manipulação em tempo real de: (i) Controles de Segurança (mecanismos utilizados para mitigar os riscos encontrados em um contexto); e (ii) Políticas de Adaptação usadas para determinar quais os controles de segurança são adequados para cada contexto.

Este trabalho apresenta o estado da arte de segurança adaptativa e está organizado da seguinte forma: o capítulo 2 apresenta os conceitos básicos, os capítulos 3 a 6 apresentam resumos de trabalhos relacionados com o tema dessa monografia e por fim, o capítulo 7 descreve a conclusão.

2 CONCEITOS BÁSICOS

Para facilitar o entendimento dos trabalhos apresentados, serão fornecidos conceitos básicos relativos a contexto, ciência de contexto, adaptação, segurança, políticas de segurança, controles de segurança e segurança adaptativa.

2.1 CONTEXTO

A noção de contexto pode ser usada para diferentes perspectivas, servindo diferentes propósitos, em diferentes áreas. Devido a essa diversidade, existem várias definições de contexto.

Schilit & Theimer em [34] referem-se a contexto como localização, identificação de pessoas e objetos ao redor e alterações nesses objetos. Em uma definição similar, Brown et al. em [24] afirmam que contexto descreve a localização, pessoas ao redor do usuário, hora do dia, estação, temperatura, entre outros.

Schilit et al. em [35] definem contexto como localização, identificação de pessoas ao redor, situações sociais, situações ambientais como iluminação e barulho, dispositivos acessíveis, capacidade e conectividade da rede e custos de computação. Segundo eles, contexto possui três aspectos importantes: (i) onde o usuário está; (ii) com quem o usuário está e (iii) quais os recursos próximos a ele.

Outras definições simplesmente usam sinônimos, referindo-se a contexto como situação ou ambiente. Dey & Abowd em [28] alegam que estas definições são muito específicas, e difíceis de serem aplicadas na prática. Segundo eles, contexto é tudo que envolve uma situação relevante para uma aplicação e seu conjunto de usuários, sendo difícil enumerar quais aspectos de todas as situações são importantes, porque eles irão mudar de situação para situação. Baseado neste argumento, Dey em [29] elaborou uma nova definição, que vem sendo utilizada em diversos trabalhos: *Contexto é qualquer informação que pode ser usada para caracterizar a situação de uma entidade. Entidade é uma pessoa, lugar ou objeto que é considerado relevante para a interação entre um usuário e uma aplicação, incluindo o usuário e a própria aplicação.*

Segundo Dey & Abowd em [28] e Dey em [29] esta definição permite que o desenvolvedor da aplicação decida quais informações podem ser contexto, em um determinado cenário. Eles afirmam que se uma parte da informação pode ser usada para caracterizar a situação de um participante de uma interação, então esta informação é contexto.

Para determinar qual informação é importante, são utilizadas as dimensões quem (who), onde (where), o quê (what) e quando (when), relacionadas a alguma entidade, para determinar o porquê (why) de uma determinada situação estar ocorrendo. Troung et al. em [37] mencionam ainda a dimensão como (how), relacionada aos métodos de captura e acesso utilizados.

Existem certos tipos de contexto que são, na prática, mais importantes que outros: localização, identidade, atividade e tempo. São chamados de contexto primário, pois não somente respondem às questões onde, quem, o quê e quando, como também podem ser usados como índices para outras fontes de informação contextual. Por exemplo, dada a identidade de uma pessoa, pode-se adquirir uma série de informações relacionadas, como endereço, telefone, e-mail, relacionamentos com outras pessoas no ambiente, etc. Através da localização de uma entidade, pode-se determinar o que são outros objetos ou pessoas próximas a ela, assim como o que ocorre em suas proximidades. Os exemplos mostram como as partes primárias do contexto em uma entidade podem ser usadas como índices para encontrar o contexto secundário (ex. endereço de e-mail) nesta mesma entidade [28].

Além do contexto do usuário, são reportadas outras classificações de contexto. Dix et al. em [30] descrevem as diferentes formas em que o contexto influencia a interação com os sistemas móveis, e os classifica em diferentes tipos, descritos a seguir.

O contexto de estrutura representa o estado da infra-estrutura de comunicação entre a aplicação e o dispositivo móvel, fornecendo meios para que mudanças no estado devido a falhas, entrada ou saída de dispositivos no ambiente possam ser reportadas [30].

O contexto de sistema, segundo Dix et al. em [30], deve possuir informações que permitam que um dispositivo esteja ciente da presença de outros dispositivos nas suas proximidades, e que as aplicações estejam cientes de outras aplicações oferecendo serviços. Esta conclusão está baseada em dois fatos: (i) dispositivos podem afetar o funcionamento uns dos outros, pois competem por recursos; (ii) combinações de dispositivos podem oferecer serviços mais avançados ao usuário.

O contexto do domínio fornece informações sobre a semântica do domínio da aplicação, levando em consideração os relacionamentos entre os dispositivos e seus usuários e como isso pode ser usado para determinar a natureza da interface ou serviço que está sendo apresentado [30]. Os autores destacam que um aspecto importante do domínio é o nível de segurança da informação e de privacidade em situações onde dispositivos são usados para identificar usuários e disponibilizar suas informações para outros usuários.

O contexto do ambiente informa as condições do ambiente no qual uma entidade se encontra, como luminosidade, temperatura, localização, entre outros.

2.2 COMPUTAÇÃO CIENTE DE CONTEXTO

O termo computação ciente de contexto (*Context-Aware Computing*) foi introduzido por Schilit & Theimer em [34], que o conceituaram como a habilidade que aplicações móveis possuem para descobrir e reagir a alterações no ambiente onde estão situadas. No trabalho apresentado por Schilit et al. em [35], o termo foi definido como o estudo de aplicações que se adaptam de acordo com sua localização de uso, com a coleção de pessoas e objetos (hosts, dispositivos acessíveis) nas proximidades e com as mudanças ocorridas com esses objetos ao longo do tempo.

Segundo Dey & Abowd em [28], um sistema é ciente de contexto se ele utiliza o contexto para prover informações ou serviços relevantes ao usuário, onde a relevância depende da tarefa do usuário. No que concerne à aplicação, a mesma é considerada ciente de contexto se usar o contexto para prover serviços ou informações ao usuário

Segundo Schilit em [36], as ações realizadas pela aplicação ciente de contexto podem ser visualizadas como um processo constituído por 3 etapas, descritas em seguida.

A primeira fase é a descoberta, que consiste no aprendizado sobre as entidades e suas características. Quando o contexto muda, o sistema tem que descobrir quais novos recursos estão disponíveis e quais são suas características e capacidades. Esta informação de contexto deve ter a representação, ser capaz de inspecionar e propagar as alterações às aplicações.

O segundo passo é a seleção, ou seja, a decisão de qual recurso utilizar. O sistema deve ser capaz de selecionar entidades baseado no contexto que o cerca. Por exemplo, onde o usuário está localizado, e quais objetos estão presentes paralelamente. Como a decisão de seleção é expressa por tipo, localização, etc., é uma preocupação fornecer uma grande quantidade de informações disponíveis no ambiente.

A última fase é o uso, que consiste no emprego do recurso, podendo ou não requerer envolvimento da aplicação.

Estas etapas requerem uma coleção bem definida de informações representando as entidades no ambiente e suas características.

2.3 ADAPTAÇÃO DE SOFTWARE

A adaptação é uma disciplina imposta pelos novos desafios da Engenharia de Software para a construção de sistemas complexos e flexíveis de forma eficiente.

Kell em [13] definiu adaptação como qualquer processo que altere seu próprio comportamento para melhorar suas interações, em resposta a alterações no ambiente.

No que diz respeito à adaptação de software, esta é caracterizada pela modificação ou extensão do comportamento dos componentes através do uso de componentes especiais, chamados adaptadores, que permitem a interoperabilidade entre componentes com interfaces incompatíveis [19].

Esses componentes são construídos automaticamente a partir de uma descrição abstrata de como as diferenças entre eles podem ser resolvidas (por exemplo, mapeamento da adaptação), baseadas na descrição das interfaces.

A adaptação ainda pode ser classificada segundo vários critérios. Um deles está relacionado ao ponto onde a mesma ocorre no ciclo de vida do software. No trabalho apresentado por Canal et al. em [20] são introduzidas duas categorias diferentes: adaptação estática e adaptação dinâmica.

A adaptação estática inclui todas as adaptações realizadas antes do sistema ser executado. Ela pode se referir tanto à adaptação de requisitos (ou modelo) quanto à de partes de códigos já existentes. A primeira é necessária quando a especificação de um sistema tem que ser estendida para cumprir novos requerimentos, modificar ou trocar requerimentos antigos. Um exemplo prático é a introdução de suporte para um novo protocolo de comunicação. A adaptação de partes de códigos já desenvolvidas é realizada tendo em mente o reuso do código. Um cenário típico deste tipo de adaptação é a alteração da maneira com que os serviços são requeridos, combinando a assinatura correta da entidade que oferece tais serviços. Em ambos os casos, os passos a serem realizados com a adaptação são conhecidos e foram bem planejados antes da adaptação acontecer.

As adaptações dinâmicas são realizadas em tempo de execução, situações comuns em computações ubíquas e móveis, onde os componentes a serem adaptados são desconhecidos antes do instante em que a adaptação ocorre.

Canal et al. em [20] afirmam que através da construção de adaptadores, é possível realizar a adaptação para tratar diferentes níveis de interoperabilidade dos componentes. Os diferentes níveis são: assinatura, comportamental, semântico e de serviço.

No nível de assinatura, as descrições de interface especificam os métodos ou serviços que a entidade oferece. Porém, os nomes dos serviços requeridos ou oferecidos podem ser diferentes, ou pode haver a necessidade de reordenar parâmetros, para adaptar seus tipos, ou para sintetizar alguns parâmetros que faltam.

No nível comportamental, as interfaces especificam o protocolo descrito no ambiente interativo que um componente segue, e também o comportamento que é esperado pelo ambiente. Todavia, pode não haver uma correspondência um-para-um em cada um dos serviços requeridos (em vez disso, um dos componentes pode considerar como um único serviço o que será obtido invocando vários serviços em outro componente), ou os protocolos seguidos pelos componentes podem ser incompatíveis, requerendo transformação de protocolo.

No nível semântico, as possibilidades de adaptação, especialmente automática, são menos evidentes. Este nível trata, por exemplo, de situações em que se busca por um componente que realize uma determinada tarefa (descrita em sua interface) e encontra-se um componente oferecendo uma funcionalidade parecida; ou ainda de situações onde não existe um único, mas sim vários componentes que atendem a tarefa desejada, envolvendo, portanto a composição adequada dos componentes encontrados.

Já no nível de serviço, existe a necessidade de adaptar o componente encontrado para obter Quality of Service - QoS, ou qualidade de serviço, e de usar vários componentes em paralelo para atingir baixos tempos de resposta, ótima precisão ou tolerância a falhas.

Entretanto, para que se possa empregar a adaptabilidade no desenvolvimento de software é necessário conhecer as maneiras de se fazer adaptação. Carvalho et al. em [21] destacam 5 modelos de adaptação.

O primeiro modelo, a adaptação à descrição “é a capacidade de descrever uma aplicação permitindo sua adequação ou transformação em diferentes linguagens ou plataformas de programação. Nesse caso, a aplicação é descrita em uma meta-linguagem e é transformada em uma linguagem de programação (e.g., Java e C++) ou em uma linguagem de marcação de hipertexto (e.g., XHTML e WML)” [21].

A adaptação ao dispositivo “é a habilidade que uma aplicação possui de adequar seu modo de execução às características do dispositivo. Essas características podem ser estáticas, como número de cores e dimensões da tela, e dinâmicas, como a quantidade de memória de execução disponível.”

A adaptação ao contexto “é a propriedade de uma aplicação de adequar-se a mudanças no contexto em que executa. As mudanças no contexto podem ser decorrentes, por exemplo,

das alterações da localização do dispositivo, do interesse do usuário e da largura de banda de comunicação".

Segundo os autores, é importante salientar que a adaptação não está restrita apenas ao comportamento da aplicação em função do ambiente de execução alvo, mas também à adequação dos dados que ela pode acessar. Nesse caso, o Móbile Application Servir - MAS são responsáveis por realizar a adaptação do conteúdo direcionado para os dispositivos. Esta maneira de fazer adaptação é denominada adaptação de conteúdo.

Quando um MAS é acessado por dispositivos móveis diferentes, deve ter seu conteúdo selecionado e adaptado às condições restritivas de cada equipamento destino (e.g., dimensões do display, quantidade de cores, memória disponível). Por exemplo, o conteúdo enviado para um notebook precisa ser diferente do conteúdo enviado a um celular, pois deve haver tanto a redução das informações, como a modificação do formato dos dados a serem apresentados. Essa adaptação de conteúdo é baseada nas características dos dispositivos, nas preferências e no contexto do usuário que utiliza a aplicação. "Algumas informações sobre a localização, o perfil e os parâmetros de qualidade de serviço do usuário são importantes para a decisão relacionada à filtragem da informação [21 apud 22]". Essa adaptação é chamada de adaptação semântica, e ainda pode levar em consideração a linguagem do usuário.

2.4 SEGURANÇA E SEGURANÇA ADAPTATIVA

Um sistema é dito seguro quando atende aos princípios básicos relacionados aos recursos que o compõem: integridade, confidencialidade e disponibilidade.

A integridade é a proteção dos dados ou informações contra modificações intencionais ou acidentais não-autorizadas, a confidencialidade é a proteção de sistemas de informação para impedir que pessoas não autorizadas tenham acesso e a disponibilidade é a garantia de que os serviços ou recursos do sistema estão disponíveis sempre que solicitados.

Alguns autores consideram outros princípios como a autenticidade, a irrevogabilidade e a legalidade. A primeira consiste no processo de verificação da identidade do usuário; a segunda é a garantia de que o emissor de uma mensagem ou uma pessoa que executou determinada transação não negue sua autoria posteriormente e a última diz respeito ao estado legal da informação, em conformidade com os preceitos da legislação em vigor.

Com o objetivo de determinar a postura de uma organização em relação à segurança, há um conjunto de regras que devem ser seguidas pelos utilizadores dos recursos

denominadas políticas de segurança. Uma política de segurança define as ações seguras e inseguras, bem como controles de acesso aos recursos e autorização.

Ainda em relação à segurança da informação, são mencionados três controles ou categorias principais que ajudam na implementação de uma segurança apropriada. O primeiro é o controle físico, usado para evitar o acesso não autorizado ao material físico, por exemplo, câmeras, alarmes, portas de aço trancadas e biometria. O segundo é o controle técnico, que utiliza a tecnologia para controlar o acesso e o uso de informações através da rede, como por exemplo, autenticação e criptografia. O último controle é o administrativo, referindo-se aos fatores humanos da empresa, envolvendo os níveis de pessoal e a determinação de quais usuários tem acesso a quais recursos e informações.

Conforme mencionado na Seção 2.3, uma aplicação se adapta de acordo com a mudança do contexto. Da mesma forma, a segurança adaptativa busca adaptar dinamicamente o conjunto de controles de segurança durante a execução de aplicações ubíquas nos dispositivos móveis, de forma a garantir a disponibilidade, a confidencialidade e a integridade dessas aplicações. Controles de segurança são definidos para cada contexto operacional e políticas adaptativas são determinadas para definir quais controles de segurança devem ser usados para um dado contexto.

2.5 CONSIDERAÇÕES FINAIS

Nesse capítulo foram apresentados os conceitos básicos, necessários para a compreensão dos capítulos a seguir, que apresentarão o estado da arte de segurança adaptativa.

3 UM LEVANTAMENTO DAS ABORDAGENS PARA SEGURANÇA ADAPTATIVA

O levantamento realizado por Elkhodary & Whittle em [1], no trabalho intitulado "*A Survey of Approaches to Adaptive Application Security*", apresenta uma abordagem de segurança adaptativa a nível de aplicação e avalia 4 abordagens atuais de segurança adaptativa conforme o esquema de avaliação introduzido no trabalho.

O esquema de avaliação proposto por Elkhodary & Whittle adota um paradigma que categoriza as abordagens de adaptação em 3 dimensões: (i) paradigma computacional; (ii) tempo de adaptação (*adaptation time*); (iii) camada de adaptação e introduz cinco novas dimensões de adaptação relevantes para a aplicação da segurança. As dimensões são agrupadas em duas perspectivas: (i) serviços de segurança adaptativa e (ii) métodos de adaptação.

O capítulo está organizado em 4 Seções, descritas a seguir. A Seção 3.1 apresenta os serviços de segurança adaptativa, a Seção 3.2 apresenta os métodos de adaptação, a Seção 3.3 avalia quatro abordagens de segurança existentes conforme o esquema proposto. Por fim, a Seção 3.4 descreve as considerações finais.

3.1 SERVIÇOS DE SEGURANÇA ADAPTATIVA

A perspectiva dos serviços de segurança adaptativa identifica um conjunto de serviços de segurança que devem ser implementados pelos sistemas, a fim de satisfazer os objetivos de segurança do software. Segundo [1], é preciso não só atentar para a presença dos serviços, mas também para o fato de eles serem "adaptativos" (figura 1). Ou seja, quando certos eventos mudam o nível de ameaça do sistema, serviços de segurança devem se adaptar em conformidade a fim de manter a satisfação dos objetivos de segurança. As dimensões de segurança são compostas por seus três principais serviços: (i) autenticação, (ii) autorização, e (iii) tolerância.

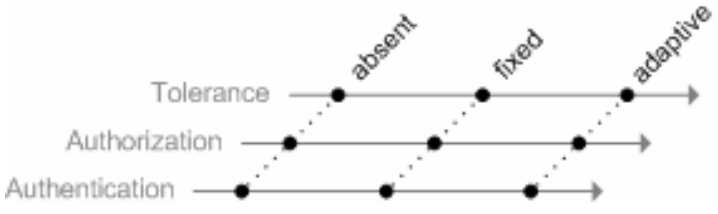

Figura 1: Dimensão da perspectiva dos serviços de segurança

A autenticação adaptativa permite que um sistema escolha entre os diferentes métodos de autenticação baseado em parâmetros específicos do contexto em tempo de execução, tais como ameaças ao sistema ou nível de suspeita do usuário. O método de autenticação pode mudar a qualquer momento, de acordo com os valores dos parâmetros de interesse.

A autorização adaptativa permite que políticas de controle de acesso sejam alteradas baseadas em parâmetros específicos de contexto. Os parâmetros podem ser globais do sistema, específicos de usuário, ou ambos.

A tolerância adaptativa é o processo de mascaramento de falhas do sistema durante a execução, eliminando componentes defeituosos, limitando os efeitos da falha e alocando substitutos válidos [6], [7] e [8].

3.2 MÉTODOS DE ADAPTAÇÃO

A perspectiva do método de adaptação examina o mecanismo de (re)configuração empregado pelo sistema adaptativo em três dimensões, (i) paradigma computacional, (ii) escala de reconfiguração, e (iii) tratamento de conflitos (figura 2).

3.2.1 **Paradigma Computacional**

Paradigma computacional, introduzido pela primeira vez em [2], é o mecanismo de transformação pelo qual as unidades de reconfiguração são inicializadas ou compostas em tempo de execução, para passarem o sistema de uma configuração válida para outra. O levantamento em [2] define quatro principais paradigmas para a reconfiguração, descritos a seguir

a) Parametrização - capacidade de mudar o comportamento de uma unidade de configuração em tempo de execução, com base no parâmetro de contexto ou usuário selecionado. A abordagem é adequada quando o número de unidades de configuração é pequeno e todas as configurações de sistema possíveis são conhecidas no tempo projetado. As principais desvantagens desta abordagem são o suporte limitado para composições complexas e o grande número de configurações. Além disso, todas as configurações devem ser rigidamente codificadas durante o tempo projetado.
b) A composição baseada em componentes - capacidade de reconectar diferentes implementações de uma interface de componentes bem definida em tempo de

execução e sem recompilação. Esta abordagem suporta qualquer configuração do sistema, desde que satisfaça os requisitos especificados pelas interfaces. Contudo, o paradigma não suporta alteração da implementação interna do componente.

c) Reflexão - capacidade de um programa de observar e modificar o seu próprio comportamento, revelando alguns detalhes de sua implementação. Neste paradigma, é possível realizar mudanças na implementação interna de um componente em tempo de execução. Além disso, esta mudança não é delimitada por requisitos como na composição baseada em componentes. Assim, pode ser realizada uma gama muito ampla de reconfigurações não esperadas.

d) Orientação a aspecto - permite a separação de características funcionais do programa a partir de suas características transversais (*crosscutting*). Em tempo de execução, todas as características podem ser compostas baseadas em um cruzamento de pontos de execução (*join point)*. Assim como a reflexão, a orientação a aspecto também suporta um vasto leque de reconfigurações. E, além disto, simplifica o desenvolvimento e a manutenção do sistema de forma significativa.

3.2.2 **Escala de Reconfiguração**

A escala de reconfiguração refere-se à escala na qual a reconfiguração pode ser alcançada (unidade única, inter-unidade, largura de sistema). O limite inferior desta dimensão é a reconfiguração da unidade única, o que permite adaptação do comportamento. O limite superior é a reconfiguração do escopo da arquitetura (*architecture-wide*), que envolve reestruturação de todas as configurações dos componentes do sistema, bem como mudanças nas propriedades externas da arquitetura do sistema.

3.2.3 **Tratamento de conflitos**

O tratamento de conflitos refere-se à capacidade do sistema de detectar e resolver as inconsistências que surgem devido às incompatibilidades entre as unidades de configuração e aos conflitos naturais de seus objetivos. A detecção de conflitos é um requisito de reconfiguração dinâmica essencial. É muito provável, especialmente para configurações baseadas em tempo de execução, a existência de conflitos estruturais ou comportamentais. Tais conflitos podem gerar graves conseqüências para o sistema e, portanto, devem ser detectados antes da reconfiguração ser executada.

A resolução do conflito é outro requisito importante que permite ao sistema adaptativo corrigir problemas detectados em tempo de execução. Três abordagens para esta resolução podem ser seguidas (a detecção deste conflito é sempre um pré-requisito).

a) Resolução de conflitos impulsionada pelo usuário - dá aos engenheiros o controle completo sobre o processo de resolução. O sistema não faz quaisquer decisões em nome do engenheiro, podendo ser necessário o trabalho manual.
b) Resolução autônoma dos conflitos - permite ao sistema decidir todas as resoluções, aliviando os engenheiros de fazê-las manualmente, mas em alguns casos, não é intuitiva nem flexível.
c) Resolução Interativa dos conflitos - mistura vantagens das duas primeiras abordagens. Ele trata as resoluções triviais automaticamente e interage com engenheiros quando são encontradas decisões mais complexas.

Figura 2: Dimensões da perspectiva dos métodos adaptativos

3.3 AVALIAÇÃO DAS PROPOSTAS DE SEGURANÇA ADAPTATIVA

Com base neste esquema de classificação da segurança da aplicação adaptativa, quatro abordagens de segurança adaptativa encontradas na literatura foram avaliadas por [1]. As abordagens são (i) Infra-estrtutura de segurança ampla, (ii) API de segurança STRATA, (iii) Arquitetura Willlow, (iv) ATNAC – *Adaptive Trust Negotiation Framework*.

3.3.1 **Infra-estrutura de segurança ampla**

Segundo Hashii et al. em [4], infra-estrutura de Segurança ampla é uma estrutura (*framework*) reconfigurável que suporta políticas de controle de acesso dinâmicas. Sua arquitetura compreende dois subsistemas principais. O primeiro subsistema é uma linguagem de especificação de política que pode ser usada para definir políticas de controle de acesso como uma 3-tupla de:

(1) Eventos - operações de acesso específico que pode ser monitorado;

(2) Condições - restrições sob quais eventos requerem ações específicas;

(3) Respostas - Ações a serem realizadas antes ou depois de um evento se a condição for satisfatória.

Usuários podem escrever arquivos de política usando esta linguagem e carregá-los no sistema. Então, o subsistema de especificação de políticas analisa os arquivos e gera na memória objetos da política que estejam em conformidade com um modelo de *meta-policy* (Figura 3).

O segundo subsistema é uma API *meta-interface* que permite aos programas privilegiados a alteração das políticas de controle de acesso em tempo de execução. A API utiliza reflexão para adicionar condições e ações aos métodos dos objetos que estão na memória.

Para incorporar um controle de acesso em uma aplicação de software, os usuários da estrutura devem desenvolver classes empacotadas para os recursos do sistema. Os empacotamentos têm que ser subtipos de objetos de política e devem fornecer uma implementação concreta dos métodos herdados. Todos os objetos que sejam declarados como subtipos dos objetos da política são reconfiguráveis pela API *meta-interface* em tempo de execução.

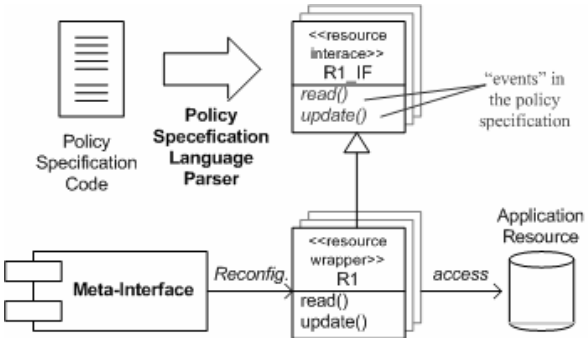

Figura 3: Estrutura da infra-estrutura de segurança ampla

De acordo com a avaliação de Elkhodary & Whittle em [1], a infra-estrutura de segurança ampla oferece autorização adaptativa permitindo que os usuários privilegiados manipulem objetos de política em tempo de execução e alterem as regras de autorização. Contudo, não oferece suporte à autenticação nem à disponibilidade.

Nesta estrutura, reflexão é o principal paradigma de adaptação. Usando a API *meta-interface*, métodos de segurança críticos são reconfiguráveis em tempo de execução. Porém a escala de reconfiguração é muito limitada. Apenas o comportamento da unidade única da política de controle de acesso pode ser configurado dinamicamente porque o sistema funciona baseado por objeto.

Além disto, os conflitos são tratados de forma autônoma. Antes de gerar objetos de política, o componente de especificação de política elimina conflitos baseados em níveis de prioridade estáticos associados aos comandos da linguagem de política.

3.3.2 **API de segurança STRATA**

Strata é uma plataforma de software que permite a modificação de um programa em execução na base de instrução. Ou seja, certas instruções binárias podem ser controladas pela plataforma e criar ouvintes correspondentes registrados pelo usuário para serem executados. Como uma implementação da tecnologia SDT, *Software Dynamic Translation*, Strata foi utilizada com sucesso em várias aplicações, incluindo segurança de software e otimização dinâmica [9], [10], [11] e [12].

Segundo Hashii et al. em [4], foi desenvolvida uma API de segurança de software para Strata, que quando utilizada, as políticas adaptativas e dinâmicas podem ser definidas e os usuários podem especificar como as regras de política alteram durante a execução quando determinados eventos (chamadas de sistema operacional e chamadas de função genéricas) são chamados por um programa em execução. Intuitivamente, políticas de segurança na Strata especificam qual código de política executar quando são invocadas as chamadas monitoradas.

Durante a tradução, quando uma função ou chamada de sistema é atingida, Strata substitui a instrução binária correspondente por uma chamada para o código de política definido pelo usuário. O código tem acesso a todos os parâmetros da função original. No entanto, Strata pode entrelaçar o código de política para otimizar a execução do programa.

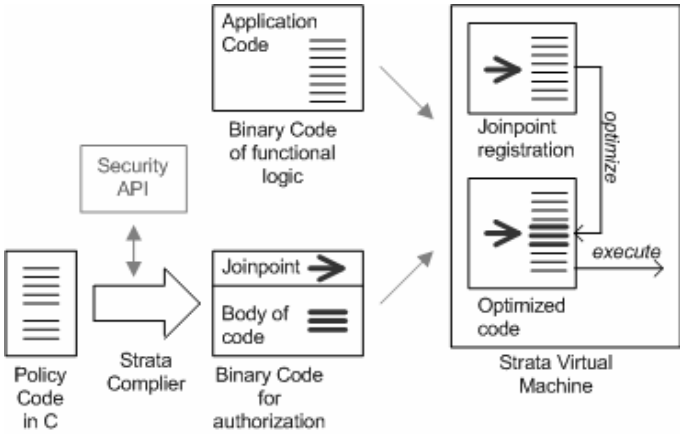

Figura 4: *Workflow* na Arquitetura Strata

De acordo com a avaliação Elkhodary & Whittle em [1], a API de segurança Strata fornece um mecanismo muito simples para implementação de políticas de autorização adaptativa. Usuários definem as políticas no nível de sistema indicando ações a serem tomadas quando determinados chamadas de sistema são invocadas. Autenticação e tolerância não estão no âmbito da API. No entanto, ela pode ser estendida para incorporar pelo menos a autenticação.

O principal paradigma apoiado pela Strata para compor códigos seguros é a orientação a aspectos.

Em relação à escala de reconfiguração, a unidade única é nativa na API. Além disso, componentes podem adicionar e remover ligações a partir de outros componentes relacionados de alguma forma.

O tratamento de conflitos não é discutido na API. Por exemplo, não está claro o que aconteceria se duas políticas monitorassem a mesma chamada de função e solicitassem ações inversas (executar versus ignorar).

3.3.3 **Arquitetura Willow**

A arquitetura Willow é uma estrutura de controle dinâmico para aplicações críticas, de larga escala e distribuídas [13], [14], [15] e [16]. Ela se esforça para manter a sobrevivência autorizando sistemas a executarem com funcionalidades reduzidas, quando estão sob condições de ataques de segurança. A arquitetura Willow compreende seis principais subsistemas, descritos a seguir.

- Controle de loops - Cada loop pode monitorar, diagnosticar, coordenar e reconfigurar em resposta a alterações no estado da aplicação, invocando componentes dedicados para cada uma dessas atividades.
- Controlador proativo - Permite aos administradores de sistema ajustarem as propriedades do sistema e emitirem reconfigurações remotamente.
- Controlador reativo - Uma entidade de controle automática organizada como um conjunto máquinas de estado finitas, que especifica estados de sistemas errôneos e ações corretivas a serem tomadas caso o sistema entre em qualquer um dos estados de erro.
- Fiscalizador de prioridade - Uma vez que as requisições de reconfiguração são emitidas pelo controle de loops, podem surgir conflitos entre elas. Deste modo, a Willow roteia todas as requisições de reconfiguração para um "cumpridor de prioridades" central que usa prioridades de requisições pré-definidas e aplica um "workflow" de reconfiguração distribuído, baseado nessas requisições.
- Infra-estrutura de legítima defesa - Protege os mecanismos da própria arquitetura e garante que os dados usados pelos monitores do sistema são atuais e precisos.
- Comunicação - Otimiza a comunicação entre nós distribuídos utilizando um eficiente serviço de notificação de evento em uma área ampla.

Os dois principais subsistemas da arquitetura Willow são o Controle Reativo e o fiscalizador de prioridades (Figura 5). O primeiro permite uma especificação sistemática das reconfigurações válidas do sistema, usando um modelo de sobrevivência formal, que atribui um valor de sobrevivência válido para cada configuração do sistema.

O fiscalizador de prioridades cria um *"workflow"* baseado em um modelo de recursos formal que associa requisições a seus recursos. Esta informação é usada para detectar conflitos entre os pedidos e para fazer cumprir uma ordem de configuração.

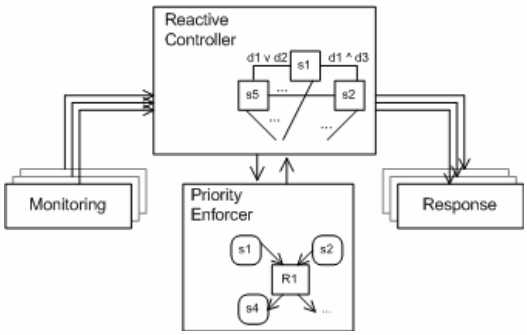

Figura 5: Os dois principais subsistemas da estrutura Willow

Segundo a avaliação de Elkhodary & Whittle em [1], o principal serviço de segurança que Willow fornece é a tolerância a falhas e intrusões.

Usando este *framework*, é possível acrescentar um aspecto adaptativo a qualquer tipo de componente, como por exemplo, incluir componentes de autenticação e autorização e modelos de reconfiguração para o conjunto integral de componentes. No entanto, os detalhes de funcionalidade de autenticação e autorização estão além do âmbito da arquitetura.

O paradigma de adaptação assumido na estrutura é a composição baseada em componentes. O modelo de reconfiguração Willow é baseado em componentes distribuídos, chamados serviços, que são tratados como caixas-pretas descartáveis, podendo ser adicionadas, removidas ou substituídas em tempo de execução. Durante a reconfiguração, se um componente for substituído, o sistema só precisa confirmar que o novo componente satisfaz o contrato original.

A reconfiguração da largura de arquitetura é nativa na Willow. Além disso, não só as interligações dos componentes são reconfiguráveis, como também as propriedades externamente visíveis da arquitetura podem ser alteradas dinamicamente pelos administradores. No entanto, o modelo de reconfiguração unidade única não é suportado. Os componentes são os melhores a nível de granularidade.

Willow trata conflitos autonomicamente utilizando modelos formais. Ele detecta conflitos entre reconfigurações baseadas no uso de seus recursos e aplica um *workflow* de composição baseado na prioridade dos pedidos de reconfiguração.

3.3.4 **ATNAC – Adaptive Trust Negotiation Framework**

Segundo [12], ATNAC – *Adaptive Trust Negotiation Framework*, ou estrutura de negociação de confiança adaptativa, é uma estrutura que permite negociação e controle de acesso adaptativo. A estrutura integra dois sistemas bem estabelecidos: GAA-API e TrustBuilder. O GAA-API fornece autorização adaptativa através de mudanças nas políticas de controle de acesso de acordo com o nível de ameaça do sistema [18]. O TrustBuilder proporciona autenticação adaptativa permitindo estabelecimento gradual da confiança com base em atributos (credenciais), exceto de identidade [14]. Os atributos são exigidos com base em um nível de suspeita associado a cada usuário (figura 6).

Para que um usuário possa acessar um recurso ou um componente específico em um domínio do fornecedor de serviços, o seguinte cenário é seguido.

(1) Usuário envia uma requisição de acesso ao provedor de serviço.

(2) Provedor de serviços consulta o GAA-API para conceder ou negar o acesso.

(3) GAA-API invoca o TrustBuilder para identificação inicial do usuário, se este for o seu primeiro pedido.

(4) Com base na política de controle de acesso e no nível de ameaça do sistema, o GAA-API decide se concede ou nega acesso.

(5) Se o acesso for concedido, o GAA-API consulta o TrustBuilder mais uma vez para requerer o conjunto de credenciais.

(6) Com base no nível de suspeita do usuário, o TrustBuilder encaminha as credenciais solicitadas necessárias para o lado cliente do TrustBuilder.

(7) Os lados cliente e servidor autenticam o usuário com base nas credenciais acordadas.

(8) Se a autenticação for bem sucedida, o GAA-API concede o acesso ao serviço solicitado.

A principal vantagem da estrutura ATNAC é a sua capacidade de determinar níveis precisos de suspeita de usuários. Ele integra heurísticas sobre cada usuário a partir de logs de controle de acesso e históricos de autenticação, o que aumenta o nível de confiança e permite que o sistema escolha as políticas menos restritivas e mais eficientes quando encontra usuários confiáveis.

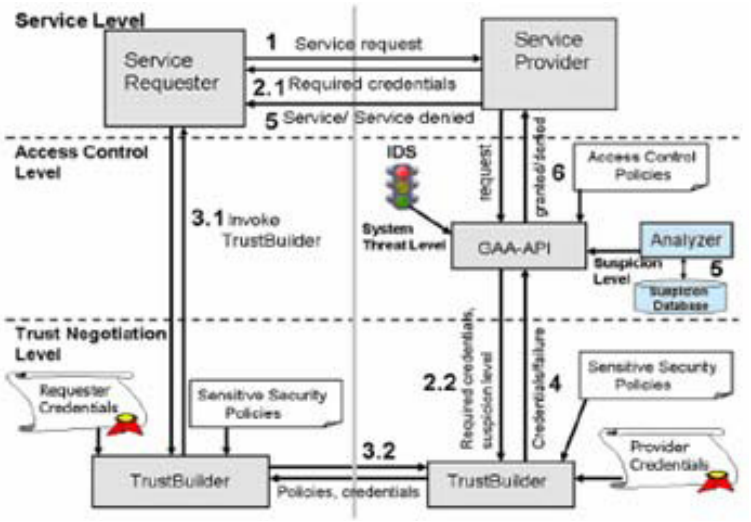

Figura 6: Cenário do típico estabelecimento de confiança na ATNAC [12]

De acordo com a avaliação de Elkhodary & Whittle em [1], a estrutura ATNAC integra os serviços de autenticação e autorização em uma única solução. Autenticação adaptativa é obtida através da associação de usuários com níveis de suspeita que aumentam ou diminuem

os pedidos de identificação no momento da execução. Já a autorização adaptativa é alcançada através da associação de políticas de controle de acesso com o nível de ameaça do sistema que restringe ou libera autorização. Apesar da ausência de mecanismos de tolerância a falhas e intrusão, ATNAC aumenta a disponibilidade de um sistema indiretamente, e usa falhas anteriores causadas pelo usuário para aumentar o nível de suspeita.

Nesta estrutura, a adaptação é baseada na parametrização. Dois parâmetros reconfiguram os componentes do sistema: nível de ameaça do sistema e nível de suspeita do usuário. Normalmente, o nível de ameaça do sistema é mantido e obtido a partir de um *Intrusion Detection System* - IDS, ou sistema de detecção de intrusos. O nível de suspeita do usuário é calculado com base em autorizações anteriores e eventos de autenticação que causaram falhas ou erros no software.

Apenas a unidade única de reconfiguração é suportada pela ATNAC. A reconfiguração é possível apenas para objetos seguros. A estrutura não aborda reconfiguração da lógica funcional, nem permite reconfiguração da largura de arquitetura. A Manipulação de conflito não está no âmbito da estrutura.

A tabela abaixo exibe um comparativo entre as abordagens para segurança de aplicações adaptativas levantadas, de acordo com o esquema proposto pelos autores em [1].

Tabela 1: Comparativo entre as abordagens atuais para segurança de aplicações adaptativas

Abordagens para segurança aplicações adaptativas	**Dimensão de Segurança**			**Dimensão Adaptativa**		
	Autenticação	**Autorização**	**Tolerância**	**Paradigma**	**Escala de Reconfiguração**	**Tratamento de Conflitos**
Infra de Segurança Ampla		Adaptativa		Reflexão	Inter-unidade	Autônomo
STRATA		Adaptativa		Orientado a aspecto	Inter-unidade	Nenhum
WILLOW			Adaptativa	Baseado em componentes	Largura de arquitetura	Autônomo
ATNAC	Adaptativa	Adaptativa	Fixa	Parametrização	Unidade única	Nenhum

3.4 CONSIDERAÇÕES FINAIS

Esse trabalho foi importante porque propôs um novo esquema de segurança adaptativa, analisando as abordagens já existentes. No capítulo a seguir, o trabalho apresentado fará uma comparação entre os ecossistemas biológicos e os ecossistemas de TI, no que diz respeito à sobrevivência e à segurança adaptativa.

4 PROJETANDO UMA ARQUITETURA DE SEGURANÇA ADAPTATIVA

O trabalho intitulado "*Designing na Adaptive Security Architecture*", apresentado por Weise em [39] introduz alguns conceitos associados à segurança adaptativa e apresenta uma abordagem que propõe o aumento da capacidade de sobrevivência do sistema, explorando as origens biológicas e ecológicas da segurança adaptativa.

O trabalho está organizado em 4 Seções, descritas a seguir. Na Seção 4.1 são introduzidos os conceitos associados à segurança adaptativa, na Seção 4.2 as diferentes propriedades biológicas e ecológicas são comparadas aos sistemas de informação, no que diz respeito à adaptação da segurança em face à uma ameaça. Na Seção 4.3, é proposta uma abordagem para implementação de uma arquitetura de segurança adaptativa, enquanto a Seção 4.4 apresenta os passos para detecção da ameaça, análise e resposta. Por fim, a Seção 4.5 descreve as considerações finais.

4.1 UMA ABORDAGEM ARQUITETÔNICA USANDO SEGURANÇA ADAPTATIVA

As metáforas biológicas e os ecossistemas fornecem paralelos interessantes para problemas, ameaças e contramedidas relacionadas a TI e à segurança do sistema. Os sistemas biológicos e ecológicos mantêm a integridade e a sobrevivência reagindo a ataques conhecidos e adaptando reações a ameaças desconhecidas.

A sobrevivência é definida como "a *capacidade de um sistema para cumprir sua missão de maneira oportuna, na presença de ataques, falhas, ou acidentes* "[42].

Segundo Weise em [39], para garantir a sobrevivência de um sistema, é necessário em primeiro lugar, identificar os elementos essenciais do sistema (o que deve sobreviver) versus os elementos que são considerados sacrificados.

Segundo Darwin, organismos se adaptam ou morrem devido aos fatores externos relacionados à imunidade humana. Fazendo uma analogia entre os ecossistemas biológicos e os ecossistemas de TI, as infra-estruturas de TI tornam-se amplamente vulneráveis, expondo-se a falhas, ataques, infecções virais, caso não se adaptem às mudanças ambientais.

Nos sistemas imunológicos humanos existem mediadores de respostas imunes (por exemplo, células-T). Esta funcionalidade é um importante componente para implementação de uma abordagem de segurança adaptativa. O papel que estes mediadores desempenham na Infra-estrutura de TI é quase idêntico ao modo como são usados no sentido biológico - eles são agentes guardiões ou sentinelas que estão implantadas em toda a infra-estrutura e agem

como sensores para identificar potenciais ameaças. Em conjunto com a resposta a ameaça e mecanismos de *feedback*, estas sentinelas moderam a resposta imune como fazem no sistema biológico. Quando é detectada uma ameaça, elas atuam como "*threat triggers*", identificando e contatando quem responde à ameaça. Essas funcionalidades da sentinela podem ser diretamente incorporadas em diversas aplicações, rede e componentes do sistema operacional. Desta forma, IDS independentes ou sistemas de firewall já não são necessários.

Entretanto, para que os sistemas identifiquem as ameaças de forma eficaz, eles devem saber o que é considerado um comportamento normal e o que não é. Utilizando uma metáfora biológica, os conceitos de "*self*" e "*non-self*" são fundamentais para sistemas auto imunológicos. Um sistema auto imunológico funcional é capaz de discriminar adequadamente entre aquilo que é nativo no organismo e o que não é. Desta forma, o que não é nativo é tratado como uma ameaça e eliminado. Baseado nesta noção, um sistema de TI pode proteger-se identificando corretamente e combatendo as ameaças e atividades suspeitas, distinguindo-as dos componentes aceitáveis, protocolos e processos operacionais na infra-estrutura de TI.

4.2 Princípios Biológicos e da Segurança de TI

Segurança adaptativa influencia os princípios operacionais e arquitetônicos a partir de uma variedade de regras. Os seguintes princípios mapeiam diferentes propriedades biológicas e de sistemas ecológicos que são aplicáveis aos sistemas de informação [39].

- Padrão de reconhecimento

  No mundo biológico as células reconhecem várias proteínas através da correspondência padrão da superfície. Da mesma forma, os sistemas TI devem ser capazes de combinar técnicas sofisticadas para identificar os comportamentos normais e anormais no código, diálogos de comando ou resposta, protocolos de comunicação, etc.

- Unicidade

  Organismos biológicos possuem seu próprio sistema imunológico que varia de indivíduo para indivíduo. Esta singularidade está associada a pontos fortes e fracos individuais. Em sistemas de TI, unicidade desestimula a existência de monoculturas que podem ser propensas a vírus comuns. A unicidade é dotada de um ecossistema de diferentes sistemas de TI com a robustez necessária para sobreviver às ameaças.

- Auto-identidade

  A noção de “self” e “non-self” permite a um organismo compreender o que é nativo e o que não é, e desencadeia a eliminação de entidades “non-self” que são consideradas ameaças. O mundo de TI imita este conceito, isolando e eliminando as ameaças, de acordo com a política de segurança. Parte da implementação deste conceito inclui compartilhamento de informações sobre ameaças, contramedidas, políticas de segurança e relações de confiança entre os sistemas de TI e infra-estruturas.

- Diversidade

  No mundo biológico refere-se à diversidade dos diferentes tipos de elementos (proteínas, células, etc.), que juntos apresentam uma vasta gama de defesas contra diversas ameaças, incluindo as respostas imunológicas adaptativas e inatas. Nos sistemas de TI a diversidade se manifesta através de diferentes mecanismos de controle, tais como compartimentalização via virtualização do sistema operacional em módulos de plataformas confiáveis, Trusted Platform Module – TPM, baseados em âncoras de hardware confiáveis.

- Disponibilidade

  No que diz respeito ao sistema imunológico, refere-se à noção de que nenhuma única célula ou molécula é essencial para todo o organismo. Um sistema de TI sacrificado - um sistema ou instancia de máquina virtual que pode ser eliminado se necessário - representa o conceito de disponibilidade em uma infra-estrutura de TI. Disponibilidade permite flexibilidade que contribui para a total robustez da infra-estrutura.

- Autonomia

  Em sistemas biológicos, significa que não existe um único elemento que controla o sistema imunológico - diferentes elementos do sistema imunológico funcionam autonomamente para combater ameaças. Na infra-estrutura de TI, é igualmente desejável que a segurança e mecanismos de controle de integridade funcionem no modo autônomo para enfrentar ameaças.

- Multicamada

  Em organismos biológicos, moleculares e celulares, outros elementos cooperam para fornecer uma capacidade de resposta à ameaça. Isto é semelhante à noção de “defense-in-depth”, que é sustentada por uma arquitetura de segurança bem projetada, e implementa várias medidas de segurança para superar os riscos de uma única medida de compromisso.

- Camada não segura

  No mundo biológico, quaisquer e todas as células em um organismo se encontram em risco de serem atacadas a qualquer momento. Esta realidade possui um paralelo exato no mundo de TI e é o pressuposto subjacente a qualquer política de segurança. Este pressuposto é instanciado através de uma política de segurança de "negar tudo", que concede acesso em apenas em casos de necessidade, e é o fundamento do "*defense-in-depth*", abordagem na qual múltiplas camadas de segurança são implementados para fornecer uma abrangente estratégia defensiva.

- Detecção irregular

  Em sistemas imunológicos biológicos, a noção de "*non-self*" permite ao sistema reconhecer e responder a entidades externas. Do mesmo modo, um sistema de TI deve reconhecer e responder automaticamente diante de um comportamento anormal ou da presença de ameaças conhecidas. A intenção de utilizar uma abordagem adaptativa para concepção de segurança é a de antecipar ameaças antes que elas se manifestem.

- Alteração do tratamento dinamicamente

  Sistemas imunológicos biológicos têm limitações em relação ao número e tipo de células e moléculas que podem detectar e reagir a patogenias. Em uma infra-estrutura de TI, existem limitações semelhantes para o número de assinaturas de ameaças e mecanismos de resposta às ameaças. Assim, uma concepção de segurança deve incluir um meio para prever e antecipar como os mecanismos de resposta às ameaças devem se posicionar a qualquer momento.

- Distributividade.

  Diferentes elementos do sistema imunológico biológico são amplamente distribuídos por todo o organismo e não estão sob qualquer controle centralizado. Em uma infra-estrutura TI esta distributividade reduz a superfície de ataque, estreitando a exposição.

- Tolerância a ruídos

  Sistemas imunológicos biológicos não exigem uma combinação absoluta para reconhecer patogenias. No mundo de TI, também é necessário que as ameaças sejam reconhecidas sem uma combinação absoluta para suas assinaturas.

- Resiliência

  Embora várias condições possam reduzir a eficácia de sistemas imunológicos biológicos, o sistema mantém um nível de resiliência que o permite a continuar a

reconhecer e a lutar contra patogenias. Um sistema deve ser resistente para que ele continue a funcionar, mesmo com capacidade reduzida.

- Tolerância a falhas

  Sistemas imunológicos biológicos são compostos de elementos redundantes que funcionam de forma complementar. Além disso, os elementos podem ser modificados para responder a patogenias as quais normalmente não responderiam. Em uma infra-estrutura de TI, tolerância a falhas permite que alguns mecanismos de resposta a ameaças sejam revisados ou modificados para responder a ameaças que normalmente não teriam respostas.

- Robustez

  No mundo biológico robustez é definida como um benefício agregado a diversidade e distributividade. No mundo de TI, faz sentido que a infra-estrutura e os sistemas também apresentem robustez.

- Memória e aprendizagem imunológicas

  No mundo biológico o sistema imunológico é por definição, adaptativo à natureza. Esta adaptabilidade permite respostas mais rápidas e mais eficazes que melhoram ao longo do tempo assim como o sistema imunológico, que aprende e retêm na memória as patogenias. Em uma arquitetura de segurança, é desejável essa adaptabilidade e, em particular, memorizar ameaças ao longo do tempo. Muitos vendedores de software para detecção de malwares atualizam suas assinaturas de dados ao longo do tempo por este motivo.

- Padrão de resposta predador-presa

  Sistemas biológicos imunológicos respondem a patogenias através de um mecanismo mediador de respostas. Isto lhes permite dimensionar a resposta à medida que o número de agentes patogênicos aumenta. Esse mecanismo de resposta mediada é igualmente necessário em um ambiente de TI, para suportar o nível adequado de controle de resposta a ameaça. Para prover essa mediação, são utilizados frequentemente mecanismos de desencadeamento e *feedback*.

- Auto-organização

  Um sistema biológico não predetermina como ele vai reagir a um desafio, mas se lembra de como ele responde. Ele determina a resposta mais eficaz e, em seguida mantém os elementos que contribuíram para aquela resposta ao mesmo tempo em que expulsam outros elementos. Em uma abordagem adaptativa, todos os controles de

resposta a ameaças devem ser capazes de ajustarem seus comportamentos de forma semelhante, para que possam empregar as contramedidas mais eficazes.

- Integração com outros sistemas

  Organismos biológicos são compostos por muitos sistemas que podem ser utilizados independentemente ou em conjunto dentro de um ecossistema maior. Em uma arquitetura de segurança, a estratégia "*defense in deph*" contribui para que sistemas de TI atinjam um nível de flexibilidade semelhante. A adesão ao referido conjunto de princípios arquitetônicos pode ajudar no desenvolvimento de um projeto de segurança eficaz. O paralelismo entre os sistemas adaptativos imunes pode ajudar a moldar os objetivos e características de um modelo adaptativo de segurança de TI.

4.3 PROJETANDO UMA ABORDAGEM PARA UM MODELO DE SEGURANÇA ADAPTATIVA

Considerando o sucesso global da maioria dos ecossistemas biológicos, imitar um sistema autonômico que incorpora uma capacidade de resposta imune é uma abordagem razoável na criação de uma infra-estrutura segura. A principal dificuldade é criar um sistema que não replique a complexidade dos sistemas biológicos, mas apóie a diversidade de sistemas e resista ao uso de monoculturas.

Um exemplo simples de aplicação de alguns princípios adaptativos é uma arquitetura de segurança "*defense-in-deph*" que como citado anteriormente, implementa várias estratégias diferentes. A diversidade pode ser alcançada utilizando mecanismos como clusterização, redundância de hardware, ou vários tipos de firewall, de múltiplos fornecedores. Desta forma, se um componente sucumbe a uma ameaça em especial, é provável que os outros não, por isso a sobrevivência do sistema é mantida. Do mesmo modo, a propriedade de resiliência pode ser sustentada através de técnicas de virtualização. Usando a tecnologia de virtualização, sistemas de infra-estruturas podem compartimentalizar diferentes serviços de sistema na execução segura dos recipientes. Tais recipientes podem ser usados para isolar instâncias de serviço. Se uma ameaça afeta um serviço em um recipiente, isto não afeta a execução dos serviços em execução em outros recipientes, de forma que os serviços dentro da infra-estrutura possam continuar. Ao mesmo tempo, mecanismos de resposta podem isolar o recipiente afetado e localizar o impacto do ataque.

O principal diferencial de uma arquitetura de segurança adaptativa é que medidas de segurança adaptativa são destinadas a proteger não só contra ameaças conhecidas, mas a

antecipar patogenias ou ameaças desconhecidas de uma forma semelhante ao sistema imunológico humano.

Weise em [39] esboçou uma possível abordagem para implementação de uma arquitetura de segurança adaptativa, através de uma lista de passos a serem seguidos. Ele ressalta que tal abordagem deve ser integrada em uma arquitetura global de segurança e ocorrer dentro do contexto de outros recursos de segurança tais como aplicação, sistema, projeto de rede e validação da configuração para garantir que todos os componentes e entidades projetadas cumprirão a política de segurança global. A lista de passos é descrita a seguir.

(1) Definir ameaças e suas características desejáveis a fim de evitar destruição. A característica de uma ameaça pode ser um atributo de uma ameaça conhecida, mas pode não incluir toda estrutura da ameaça. Isto também pode ser um comportamento particular exibido por algumas entidades ou processos (Este conceito está ligado à noção de "*self*" e *"non-self"*).

(2) Identificar os comportamentos aceitáveis, componentes confiáveis, e ações que não devem ser confundidas com uma ameaça. (Esta etapa é fundamental para evitar um ataque denial-of-service – DoS e também está relacionada com a noção de "*self*" e "*não-self"*)

(3) Definir "*triggers*", para monitorar ameaças e, se necessário, para invocar uma resposta do sistema auto-imunológico. Estes "mediadores de resposta imunológicos" são sensores de detecção de ameaças que alertam as maiores infra-estruturas de TI sobre as potenciais ameaças e prepara os principais mecanismos de respostas.

(4) Implementar redundância de funções críticas, evitando o comprometimento de todo o sistema. Esta implementação instancia as noções de sobrevivência, diversidade e redundância.

(5) Definir mecanismos de resposta a ameaças que recebem maior foco, evitando a morte host.

(6) Definir um processo de recuperação pelo qual um sistema é capaz de se reconfigurar e reiniciar adaptativamente. Parte deste processo também inclui uma aprendizagem e conhecimento de mecanismos de distribuição, e desta forma a infra-estrutura aprende como evitar ameaças semelhantes no futuro.

(7) Definir capacidades de feedback que permitam aos mecanismos de resposta validarem ameaças para que eles só respondem às legítimas. Os mecanismos de *feedback* ajudam

a garantir que os "*triggers*" e mecanismos de respostas compreendam o contexto de segurança em que operam, permitindo o comportamento adaptativo desejado.

Nem todo sistema é obrigado a possuir todas as características das ameaças definidas. O objetivo é criar uma população variada de sistemas onde cada um tenha diferentes capacidades de resposta e juntos facilitem a sobrevivência do ecossistema, através do compartilhamento de conhecimentos [39].

4.4 PROCESSAMENTO DE SEGURANÇA ADAPTATIVA

Segundo Weise em [39], segurança adaptativa baseia-se na noção fundamental de "*self*" e "*non-self*", onde um sistema deve ser capaz de compreender e reconhecer o que é um comportamento normal, e determinar se o comportamento anormal é uma ameaça. Segundo o autor, a abordagem fundamental da segurança adaptativa envolve a detecção da ameaça, análise e resposta, seguindo três passos.

O primeiro é a telemetria, que consiste na coleta e monitoramento da informação sobre o sistema, rede, fluxos de dados, gerenciamento de arquivos, manuais e atividades operacionais automatizadas e sobre quaisquer outras atividades que afetem a infra-estrutura de TI. A telemetria deve ser coletada em tempo real, antecipando as ameaças de forma eficaz. Todavia, o histórico de dados é útil, uma vez que é necessário construir um conjunto de informações que podem ser utilizados para a análise das tendências.

A Correlação é a avaliação dos dados da telemetria em tempo real, em conjunto com os dados históricos e uma política de segurança bem definida, levando em conta os grandes períodos de latência ligados aos ataques. Esta habilidade de correlacionar a informação em respostas é o ponto central da segurança adaptativa e existem áreas de pesquisa promissoras a respeito do assunto, como por exemplo, a predição analítica.

O último passo é a resposta, ou seja, uma ou mais ações tomadas para reagir a uma ameaça. Os mecanismos de resposta agem para impedir a introdução ou propagação de uma ameaça na Infra-estrutura de TI. Frequentemente as respostas incluem a modificação da configuração dos sistemas, de suas características e comportamentos; e em alguns casos, sua paralisação. O objetivo dos mecanismos de resposta é limitar a exposição e os impactos que podem afetar os níveis de serviço.

Estes três passos básicos - telemetria, correlação, e resposta devem ser automatizados para permitir a rápida implantação de patches e atualizações de segurança. A automatização destas etapas e a remoção do elemento humano (quando possível), tem a vantagem de reduzir

erros. Além disso, as respostas às ameaças antes que elas se manifestam plenamente visam: (i) o aumento da eficiência operacional; (ii) a limitação da superfície de ataque da infra-estrutura de TI; (iii) a redução da velocidade dos ataques e do tempo de recuperação, assim como a reabilitação do sistema.

4.5 CONSIDERAÇÕES FINAIS

Este trabalho foi importante porque propôs a criação de uma infra-estrutura adaptativa capaz de responder às ameaças de segurança, baseada em uma analogia entre os sistemas imunológicos humanos e os ecossistemas de TI, onde os IDS aprendem a combater a novos tipos de ataque. No capítulo a seguir, será apresentado um trabalho que discutirá questões relacionadas à segurança adaptativa.

5 INTRODUÇÃO AOS FUNDAMENTOS LÓGICOS DE UMA INFRA-ESTRUTURA DE SEGURANÇA ADAPTATIVA

Este capítulo apresenta um resumo do trabalho do autor Leo Marcus, intitulado "Introduction to Logical Foundations of an Adaptive Security Infrastucture" [43], que objetiva introduzir o conceito de infra-estrutura de segurança adaptativa, Adaptive Security Infrastructure - ASI e discutir questões relacionadas ao tema.

O capítulo está organizado em 6 seções, descritas a seguir. Na Seção 5.1 é introduzido o conceito de infra-estrutura de segurança adaptativa, na Seção 5.2 são apresentados os princípios da formalização de teoremas, na Seção 5.3, políticas de segurança são discutidas. Nas Seções 5.4 e 5.5, são propostos tipos de resposta defensiva e métodos de detecção de intrusão, respectivamente. Na Seção 5.6 são apresentados trabalhos futuros. Por fim, a Seção 5.7 descreve as considerações finais.

5.1 COMPONENTES PRINCIPAIS DE UMA ARQUITETURA DE SEGURANÇA ADAPTATIVA

O termo infra-estrutura foi adicionado à segurança adaptativa, obtendo-se infra-estrutura de segurança adaptativa, ASI, para indicar a abordagem que vê a segurança adaptativa como um componente integral, fundamental e funcional do sistema.

Segundo Leo Marcus em [43], para coordenar a detecção de entradas de segurança relevantes, políticas de segurança, entradas de usuário, análise e, em seguida, ser capaz de formular e executar uma resposta, é necessário isolar os três componentes conceituais: sensor, análise e resposta.

O autor cita dois exemplos: o primeiro é um sistema que está em constante monitoramento, análise e respondendo de forma a manter a segurança estável e o segundo, um sistema em evolução para satisfazer novas propriedades de segurança, levando em conta a política de segurança atual, a gravidade dos efeitos ambientais, aspectos temporais e geográficos dos ataques e respostas. Diante de um destes cenários, ele levanta as seguintes questões: Que propriedades precisam ser definidas para formular os teoremas? Que propriedades os componentes e interfaces possuem para que seus efeitos satisfaçam uma propriedade desejada?

5.2 FORMALIZAÇÃO: PRINCÍPIOS E QUESTÕES

Leo Marcus em [43] apresenta alguns princípios básicos da formalização de teoremas:

1. Utilizar um quadro lógico matemático;
2. Resumir cenários realistas;
3. Evitar preocupação com a usabilidade ou a tecnologia atual (naturalmente, a um nível mais profundo, reconhecemos que a tecnologia atual tem uma inegável, se imensurável, influência sobre a nossa imaginação);
4. Definir a longo prazo um objetivo comum, uniforme, com semânticas inter-interpretáveis para permitir especificações rigorosas e verificações de arquiteturas, propriedades e capacidades que podem se conectar a políticas, detecção, análise e resposta.

O autor parte do pressuposto básico de que a ASI existe em um mundo temporal e espacial. Se aceitarmos a natureza temporal e distributiva do sistema como um todo, temos estruturas arquitetônicas arbitrárias (padrões de conectividade, por exemplo, redes generalizadas) existentes dentro do sistema e da ASI, e estas estruturas podem ser dinamicamente alteradas. Qualquer aspecto da política, especificação, detecção, análise ou resposta pode ser considerada em uma versão relativa para qualquer estrutura que possa ser definida. Chamamos isto de *Pervasive Hierarchy Assumption* (PHA).

Segundo ele, os seguintes temas de investigação podem parecer grandiosos em seus âmbitos. Na verdade eles são, mas precisam ser divididos.

1. Quais são as semânticas adequadas de uma dinâmica e adaptável política de segurança, e como devem ser especificadas?
2. Como devemos levar em conta a natureza global-local de todos os componentes de uma ASI de acordo com o PHA?
3. Como deveremos especificar os "recursos de segurança relevantes" disponíveis para que, a qualquer momento, o analisador possa escolher uma resposta adequada?
4. Como vamos especificar as capacidades de respostas (incluindo os trade-offs?)
5. Como devemos unificar os aspectos de argumentos temporal-espacial?
6. Quais são as questões complexas do sistema?
7. Qual é o papel de uma "segurança aproximada"?

5.2.1 **Questões de pesquisa: espaciais**

Algumas das questões de pesquisas interessantes relacionadas com a dimensão espacial são as seguintes [43]:

1. Especificação das arquiteturas hierárquicas;
2. Detecção central (local) e distribuída (global), análise e coordenação de resposta;
3. Transição suave entre as hierarquias;
4. Testabilidade da política de satisfação;
5. Executoriedade da resposta.

5.2.2 **Questões de pesquisa: temporais**

Algumas questões de pesquisa relacionadas à dimensão temporal são:

1. Duração da resposta;
2. Sincronização;
3. Velocidade relativa da troca de ambiente, detecção, análise, comunicação e resposta;
4. Incorporação do tempo na política;
5. Reconhecimento.

5.3 POLÍTICA DE SEGURANÇA ADAPTATIVA

A meta para a especificação de segurança adaptativa é dupla: (i) fornecer um guia abrangente para decidir se os acontecimentos futuros, ações, ou respostas serão autorizadas pela política corrente e (ii) permitir que novas metas de segurança sejam estipuladas, a fim de dar início a um sistema de respostas para executar essa política, se necessário.

Como definido anteriormente, uma política de segurança é um conjunto de decisões que, coletivamente, determinam a postura de uma organização em relação à segurança. Mais precisamente, a política de segurança determina os limites do que é aceitável ou não e os critérios a serem adotados em função das violações. Leo Marcus defende que uma política de segurança é a especificação daquilo que é permitido. Como exemplo de uma política adaptativa, pode ser considerado o seguinte:

(1) O sistema inicialmente satisfaz a política P_1;

(2) Na primeira ocorrência da condição C, o sistema troca para a política P_2;

Isto levanta imediatamente a questão: o que significa satisfazer uma política P1 em um intervalo (de uma hora ou evento t1 para outro tempo ou evento t2)?

A resposta seria não contrariar a política, ou seja, deve haver alguma continuação da computação, ou no caso políticas não primitivas, algum alargamento do conjunto de cálculos que satisfaça explicitamente a política.

5.3.1 **Política Incremental**

Uma mudança de política é incremental quando sabemos qual aspecto queremos mudar, mas não sabemos ou nos importamos com o resto da política expressa na especificação do sistema. Por exemplo, uma mudança de direitos de acesso do usuário pode ou deve ser expressa como um incremento que afeta apenas aquele usuário. Isto levanta a questão de dependências entre as políticas que podem parecer locais: talvez a mudança de um direito de acesso de um usuário altere também os direitos de acesso dos outros usuários, através de alguma interação entre eles.

O incremento pode ser um "enfraquecimento", representado pela união de políticas anteriores com a com a nova política, ou um "reforço", representado pelo conjunto de interseção entre política anterior e a nova política.

Um incremento de política pode ser indicado por: [P; C → ($+P_1$ - P_2)], onde P_1 e P_2 são as próprias políticas, o que significa: fortalecer por P_1 e depois por enfraquecer P_2. Esse incremento poderia ser uma combinação complexa de fortalecimento e enfraquecimento.

5.3.2 **Política Local**

Seja H uma descrição de hierarquia, A uma especificação de uma ASI e P uma política, onde:

- P é local em respeito a H em A

Significando algo como:

- a satisfação de P em A é dependente apenas na satisfação de algumas políticas em todos os subsistemas satisfazendo H.

Em determinadas situações podemos definir diferentemente a localidade:

1 . “Para todas as instancias de A, existe uma política teste para P como....” ou

2. “Existe uma política teste para P, como para todas as instancias de A” ou

3. ....”em alguns subsistemas satisfazendo H”

5.4 ESPECIFICAÇÃO, DERIVAÇÃO E VERIFICAÇÃO DE RESPOSTA

Uma das questões mais desafiadoras é como especificar as respostas, suas relações com os recursos, e as suas capacidades. Alguns tipos de respostas defensivas que podem ser adequados para determinados tarefas de segurança incluem, em ordem aleatória:

- Alocar recursos (ligar ou desligar dispositivos);
- Ajustar roteamento (incluir ou excluir nós);
- Alterar direitos de acesso;
- Alterar algoritmos de criptografia, chaves, protocolos;
- Alterar rede de sensores;
- Alterar auditoria;
- Alterar a força da autenticação;
- Ajustar configurações de sistemas de detecção de intrusos;
- Instalar patches;
- Sacrificar dados ou dispositivos;
- Instalar novos hardware e software.

No contexto geral de uma ASI podemos definir uma resposta como um programa distribuído ou um algoritmo executando simultaneamente a ASI e a operação de sistemas. Claro que, intuitivamente, respostas comuns têm mais propriedades específicas, como alteração e encerramento de estado.

A fim de incorporar as respostas em um quadro formal, precisamos especificar e avaliar recursos de respostas, incluindo canais de comunicação, se necessário, resistência e localização; coordenar respostas com análise; e possuir um plano de ação adequado no tempo e no espaço

5.5 DETECÇÃO E ANÁLISE

A detecção e a análise de componentes estão intimamente relacionadas. Os mecanismos de detecção atualmente empregados incluem:

1. Vários tipos de métodos de detecção intrusão (por exemplo, assinatura e anomalia);
2. Estatísticas de rede;

3. Estatística de uso de sistemas;
4. Estatísticas de ameaças;
5. Background de dados eletrônicos.

Segundo o autor Leo Marcus, não sabemos quais tipos de informações ambientais podem ser úteis no futuro. Neste caso, para coordenação desta informação, é relevante realizar pesquisas no campo de sensores. Obviamente, a possível ligação entre a natureza dos dados recolhidos, a natureza das políticas implementadas, bem como a natureza da análise de motores, e como essas conexões podem ser adaptadas entre si, é uma questão muito abrangente.

5.6 TRABALHOS FUTUROS

Leo Marcus em [43] propõe o seguinte teorema para a verificação de uma futura ASI:
1. Para qualquer sistema *S* implementar a especificação S;
2. Para qualquer ASI *A* implementar a especificação A;
3. Para qualquer política de segurança adaptativa *P* do tipo P;
4. Para qualquer ambiente *E* satisfazer as condições E.;
Onde *S* + *A satisfaz P em E.*

Neste caso, existe um problema de arquitetura da ASI. Dados *E*, *P* e *S*, encontrar *A*; Como *E* é mais "realista", P tem de obter os pontos mais fracos, a fim de que haja alguma esperança de encontrar uma A adequada.

5.7 CONSIDERAÇÕES FINAIS

Nesse capítulo foi apresentado um trabalho que propôs a implementação de políticas e métodos de segurança visando à eficiência e eficácia de uma ASI. No capítulo a seguir será apresentado um trabalho que propõe um serviço para ciência de contexto e provisão de reação.

6 UMA ARQUITETURA DE SERVIÇO PARA CONSCIÊNCIA DE CONTEXTO E PROVISÃO DE REAÇÃO

Neste trabalho, intitulado "*A Service Architecture for Context Awareness and Reaction Provisiong*", Santos et al. [44] apresentam um serviço para consciência de contexto e provisão de reação - ARS - seguindo uma abordagem baseada em regras que proporcionam reações (notificações, serviço ou invocação de aplicações), dependendo do contexto dos usuários.

O trabalho está organizado em 6 Seções, descritas a seguir. Na seção 6.1, é apresentado o serviço de consciência e provisão de reação. Na Seção seguinte, a 6.2, sua arquitetura. Na Seção 6.3, são definidos conceitos básicos referentes aos estados das situações. Nas Seções 6.4 e 6.5 são apresentados um caso de uso doméstico e trabalhos futuros, respectivamente. Por fim, a Seção 6.6 descreve as considerações finais.

6.1 A CONSCIÊNCIA E O SERVIÇO REATIVO

A Consciência e o Serviço Reativo - Awaneress and Reactive Service – ARS, oferece suporte aos programadores adicionando capacidades de consciência de contexto para suas aplicações. Assim, os desenvolvedores não têm que lidar com monitoração, controle e gestão contextual da informação dentro das suas aplicações. Isto evita a necessidade de criação de características específicas de ciência de contexto para cada aplicação e, portanto, promove um rápido desenvolvimento. As aplicações são responsáveis apenas por registrar regras de monitoramento. Tais regras especificam que o contexto deve ser controlado e que a reação deve ser desencadeada uma vez o que contexto esperado ocorra.

Uma vez que o aplicativo cliente tenha sido subscrito à regra de acompanhamento, ARS inicia a coleta de informações contextuais. No caso da condição do desencadeamento estar contida no monitoramento das regras armazenadas, ARS procede com a fase reativa, de acordo com a reação especificada na regra. Um exemplo dessa regra específica é que quando o usuário João entra sua casa, a iluminação deve ser ligada. Neste exemplo, ARS monitora a localização de João, e quando ele entra sua casa, o serviço invoca o sistema de controle de iluminação para ligar as luzes [41].

As literaturas de Costa et al. em [40] e Ipina & Katsiri em [38] consideram que as mudanças no ambiente da aplicação são modeladas por meio de regras Evento-Condição-Ação (ECA).

6.2 A ARQUITETURA ARS

Seguindo o padrão Evento-Controle-Ação descrito em [40], os três principais elementos estão presentes na ARS como demonstrado na Figura 7.

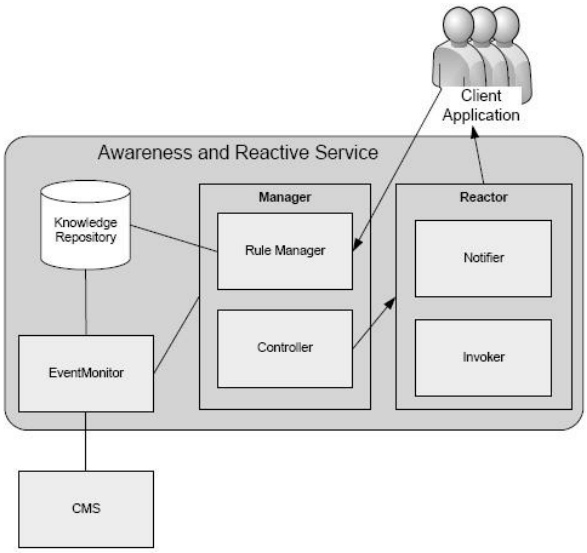

Figura 7: Arquitetura ARS

O Monitor de Eventos (EventMonitor) recebe eventos de dados de contexto a partir de fontes de contexto através do Serviço de Gerenciamento de Contexto (Context Management Service – CMS). O Monitor de Eventos envia esses eventos ao gerente que os monitora e avalia as regras registradas. Se a condição desencadeante da regra for avaliada como verdadeira, o reator é acionado para executar a ação apropriada.

As regras aprovadas e as ontologias usadas na ARS são armazenadas no Repositório de Conhecimento (KnowledgeRepository) e disponibilizadas tanto para o Gerente de Regras quanto para o monitor de eventos.

O projeto arquitetônico da ARS segue o princípio da arquitetura orientada a serviços – SOA, onde o Gerente (Manager) é o componente central da ARS e é responsável por lidar com as regras de assinaturas dos clientes, sendo composto por dois sub-componentes, o Controlador (Controller) e o Gerente de regras (RuleManager), externamente acessado através de uma interface IManageRuller, que provê facilidades para cancelar atualizações, iniciar e parar regras. Quando uma aplicação cliente quer registrar uma regra, ela envia esta regra para o Gerente de Regras, que é responsável pela análise, validação e armazenamento da nova regra. Nas fases de análise e validação, o Gerente traduz a regra dos usuários inseridas em regras de reação que podem ser tratadas pelo Controlador.

As normas recebidas pelo Gerente da aplicação cliente são expressas na ARS de domínio específico, chamadas ECA-DL [4]. O Gerente de Regras então transforma estas normas ECA-DL em uma regra manipulável pelo criador de regras.

Quando uma regra é registrada, ela está disponível para a ARS, mas ainda não está sujeita ao monitoramento, ou seja, a regra só é cadastrada no sistema, mas sua condição não vai ser avaliada. Para iniciar sua avaliação é necessário inicializá-la, e então o Gerente de regras a envia ao Controlador. Este então extrai a parte de eventos da regra e subscreve estes eventos ao Monitor de Eventos para procurar e pedir informações de fontes de contexto.

A principal funcionalidade do monitor de eventos é oferecer fácil acesso aos dados do contexto, incluindo busca de fontes de contexto.

O Monitor de Eventos provê a outros componentes ARS um mecanismo para a subscrição ou consulta de dados de contexto. Por exemplo, se o Controlador precisa acompanhar o nível da bateria de um dispositivo, o Monitor de Eventos, através do CMS, procura por uma fonte de contexto apropriada que possa fornecer essa informação. Uma vez encontrada, este subscreve o dado referente ao nível de bateria e informa ao Controlador de Eventos.

O Monitor de Eventos mantém uma assinatura da fonte de contexto correspondente para todos os eventos que o Controlador tenha solicitado. A manutenção de uma lista de eventos e fontes de contexto relacionadas é importante para evitar assinaturas redundantes. Para cumprir esta tarefa, ele analisa o pedido de assinaturas em busca de sobreposições na subscrição de requisitos. Um exemplo de exigência de requisitos é quando uma única fonte de contexto pode fornecer dois eventos diferentes. Neste caso o Monitor de Eventos mantém apenas uma assinatura para a fonte de contexto.

Depois de subscrever os eventos contidos na regra para o Monitor de Eventos, o Controlador começa a receber notificações da ocorrência destes eventos. Para cada notificação de evento recebida, o Controlador avalia as regras de notificação. Quando a condição de uma regra é avaliada como verdadeira, o Controlador invoca o Reator requisitando a reação apropriada.

6.3 CONCEITOS BÁSICOS

Mudanças de contexto são descritas como as alterações nos estados das situações. Situações representam exemplos específicos de informação de contexto e podem ser definidas como outras situações ou fatos [12].

Fatos definem o "estado de coisas" no ambiente dos usuários. Exemplo de um Fato é que Jerry é casado com Maria. A abstração da situação do contexto permite aos desenvolvedores de aplicações e aos usuários alavancarem na abstração do fato, a fim de derivar a informação de contexto de alto nível. Exemplo de uma situação é isOccupied, derivado do fato de "Jerry está cozinhando" ou "Maria está trabalhando". Situações podem ser construídas sobre outras situações, por exemplo, isAvailable pode ser definido como não isOccupied e isReachable. Fatos e situações são definidos como parte do conjunto de modelos de informação (ontologias). Um evento exprime uma mudança de estado, que é interessante para aplicações específicas. Por exemplo, uma aplicação pode estar interessada em saber quando Jerry entra na sala de TV, quando Jerry e Roberto estão juntos, ou quando Pablo fica on-line no sistema de mensagens instantâneas. Estes cenários referem-se a mudanças de estado:

- Jerry entra no quarto de TV: Estado (Jerry não está no quarto de TV), seguido pelo estado (Jerry está no quaro de TV);
- Jerry e Roberto se aproximam um do outro: Estado (Jerry e Roberto estão longe um do outro) seguido pelo Estado (Jerry e Roberto estão próximos um do outro);
- Pablo torna-se on-line: Estado (Pablo está off-line) seguido pelo Estado (Pablo está on-line).

Na ECA-DL, são permitidas as definições dos eventos de duas formas: (i) expressa por uma mudança explícita de estado, ou (ii) expressa como uma descrição de eventos. Considere os exemplos acima definidos: Podemos expressar a situação "Jerry entra sala", como EnterTrue (LocatedIn (Jerry, TVRoom)) ou EnterRoom (Jerry, TVRoom). Da mesma forma, podemos definir "Jerry e Roberto se aproximam", como EnterTrue (Aproximação (Jerry, Roberto)) ou GetClose (Jerry, Roberto). Finalmente, podemos definir "Pablo fica on-line" como EnterTrue (On-line (Pablo)) ou BecomesOnline (Pablo).

Quando se define na ECA a regra de mudança de estado utilizando as transições de estado EnterTrue e EnterFalse, por exemplo, o desenvolvedor da aplicação manifesta os eventos explicitamente definindo o estado de transição para uma determinada situação. Inversamente, desenvolvedores da aplicação podem expressar eventos usando descrições de evento pré-definidas. Isto exige que estes acontecimentos sejam previamente definidos na ontologia.

Na ECA-DL existem três estados possíveis (verdadeiro, falso e desconhecido) e seis transições de estado. O desconhecido acomoda o estado de incerteza nas informações de contexto. A Figura 9 apresenta as possíveis transições de estado.

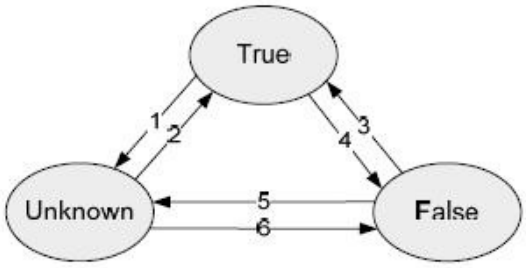

Figura 8: Transições de estado para uma situação

Para uma determinada situação S, os eventos podem ser quaisquer uns dos seguintes:

- EnterTrue(S) – transições 2 ou 3;
- EnterFalse(S) – transições 4 ou 6;
- EnterUnknown(S) – transições 1 ou 5;
- ExitTrue(S) – transições 1 ou 4;
- ExitFalse(S) – transições 3 ou 5;
- ExitUnknown(S) – transições 2 ou 6;
- TrueToFalse(S) – transição 4;
- TrueToUnknown(S) - transição 1;
- FalseToTrue(S) – transição 3;
- FalseToUnknown(S) – transição 5;
- UnknownToTrue(S) – transição 2;
- UnknownToFalse(S) – transição 6;
- Changed (S) – quaisquer transições.

A parte de condições de uma regra descreve as condições extras que devem ser prioritárias na invocação de uma notificação. Ela difere da parte do evento, uma vez que não define mudança de estado, apenas especifica requisitos adicionais (estados) que devem existir. Além de especificar um evento, desenvolvedores das aplicações podem se interessar em situações mais específicas. Por exemplo, uma aplicação pode precisar ser notificada sobre quando Jerry entra na sala de TV, sob a condição de que Jerry pode estar sozinho. A condição de que Jerry pode estar sozinho não define um estado de transição, apenas manifesta exigências adicionais para a notificação a ser invocada.

Condições são expressões lógicas que indagam se uma situação, ou uma combinação de situações são verdadeiras ou falsas. Situações referem-se a conceitos definidos na ontologia.

Ações descrevem as operações que devem ser invocadas quando as partes do evento e as condições das regras são preenchidas. Na ARS, as ações atualmente são restritas às operações de notificação e invocação. A forma como as notificações são entregues depende das preferências dos usuários e do contexto.

6.4 UM CASO DE USO LOCAL A LOCAL (DOMÉSTICO)

Santos et al. Em [44] introduz um caso de uso que é implementado no projeto Amigo [33], como uma avaliação de um número de componentes *"open source"* que serão disponibilizados pelo projeto para apoiar o desenvolvimento de aplicações inteligentes de acordo com o ambiente.

O objetivo global do conceito é que as pessoas estejam cientes da presença de seus contatos e suas atividades para iniciar a comunicação entre eles. Uma Interface tangível está sendo investigada, a "Awareness Globe", cujo protótipo é exibido na figura abaixo.

Figura 9: Protótipo da interface Awareness Globe

A figura 10 descreve esta interface, enquanto a figura 11 mostra como o dispositivo apresenta a informação.

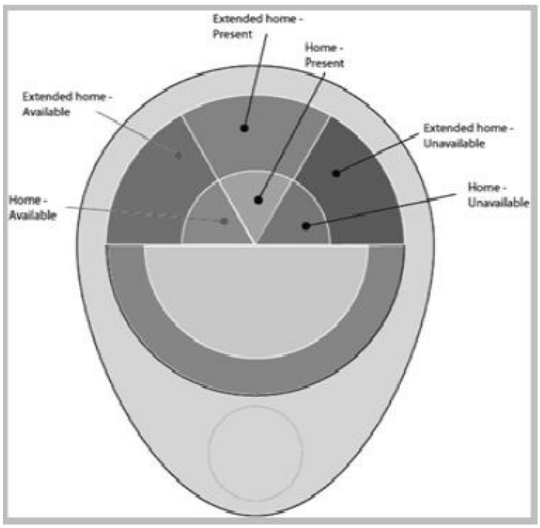

Figura 10: Apresentação das informações pelo dispositivo

A ilustração do uso da Awareness Globe é descrita pelo seguinte cenário: "Maria Volta para casa. Colocando a chave de seu carro na Awareness Globe, ela acessa os serviços e as aplicações internas e ainda pode controlar sua própria disponibilidade e presença para interagir com o mundo externo."

Um sistema de comunicação "*home to home*" integrando o ARS vem sendo atualmente desenvolvido na estrutura do Projeto Amigo [33].

6.5 TRABALHOS FUTUROS

As abordagens atuais para apoiar um middleware ciente de contexto apresentam formas para subscrever e gerenciar dados de contexto. Todavia não fornecem apoio à decisão, ou seja, não fornecem um mecanismo onde aplicações podem especificar em quais dados de contexto estão interessados e o que fazer em determinadas situações.

Como a linguagem atual de regras da ARS não fornece meios para especificar a ordem temporal dos eventos, ela será estendida para suportar os aspectos temporais, como a seqüência e concorrência de eventos, sendo considerados modelos de contexto. Além disso, a versão atual do ARS implementa localização baseada em primitivas tais como isLocatedIn, isAtHome e atualmente encontra-se em curso um trabalho de apoio a outras primitivas inteligentes em um ambiente doméstico.

6.6 CONSIDERAÇÕES FINAIS

Nesse capítulo foi apresentado um artigo que propôs a utilização de um serviço capaz de reagir conforme o contexto. No capítulo a seguir será apresentada a conclusão desta monografia.

7 CONCLUSÃO

O objetivo dessa monografia foi definir e classificar a segurança adaptativa e apresentar alguns artigos de forma a mostrar o estado da arte. Foram selecionados 4 artigos, são eles: “A Survey of Approaches to Adaptive Application Security” [1], “Designing an Adaptive Security Architecture” [39], “Introdution to logical Foundations of an Adaptive Security Infrastructure” [43] e “A Service Architecture for Context Awareness and Reaction Provisioning” [44].

O trabalho apresentado por Elkhodary et al. em [1] faz um levantamento das abordagens atuais para segurança adaptativa em nível de aplicação. Foi introduzido um esquema de avaliação a partir de duas perspectivas abrangendo a segurança crítica e requisitos de reconfiguração. Quatro abordagens foram levantadas e avaliadas de acordo com o esquema, mas nenhuma delas satisfez a gama de requisitos identificados pelo esquema.

A literatura apresentada por Weise em [39] faz um paralelo entre os organismos biológicos e ecossistemas e a concepção de uma arquitetura de segurança adaptativa para as infra-estruturas de TI. O autor defende que imitar um sistema autonômico, que inclui os mecanismos de resposta imune pode ser um instrumento para a abordagem de um projeto eficaz na construção de uma infra-estrutura que segue os princípios da segurança adaptativa. A abordagem descrita neste artigo oferece um ponto de partida para a criação de uma infra-estrutura adaptativa, que pode se ajustar e responder a ameaças potenciais de segurança. O estabelecimento de políticas de segurança e a definição de linhas de base para “self” e “non-self” são apenas os primeiros passos. A aplicação da telemetria, da correlação e dos mecanismos de resposta para processar ameaças são as chaves para o esforço de proteger os sistemas de infra-estrutura e remediar os ataques.

O trabalho apresentado por Leo Marcus em [43] introduz o conceito de infra-estrutura de segurança adaptativa e discute as questões relacionadas ao tema, como segurança e formalização lógica de políticas.

Por fim, o trabalho de Santos et al. em [44] apresenta o ARS, um serviço que habilita os desenvolvedores a rapidamente implementarem aplicações que permitam aos usuários estarem cientes de seus ambientes. As abordagens atuais para apoiar um *middleware* ciente de contexto apresentam formas para subscrever e gerenciar dados de contexto. Todavia não fornecem apoio à decisão, ou seja, não fornecem um mecanismo onde aplicações podem especificar em quais dados de contexto estão interessados e o que fazer em determinadas situações. Além disso, estas abordagens não oferecem um processo de reação baseado no

contexto do usuário. A linguagem atual de regras da ARS não fornece meios para especificar a ordem temporal dos eventos. A linguagem será estendida para suportar os aspectos temporais, como a seqüência e concorrência de eventos. Em segundo lugar, modelos de contexto serão considerados. Além disso, a versão atual do ARS implementa localização baseada em primitivas tais como isLocatedIn, isAtHome. No entanto, segundo os autores, o trabalho de apoio outras primitivas inteligentes em um ambiente doméstico está em curso.

De acordo com esses trabalhos, pude observar a importância de todo o alicerce de sustentação dos requisitos de segurança ser adaptado, pois o crescente uso de dispositivos móveis tem influenciado uma série de negócios, sendo necessário definir políticas eficazes para alcançar a segurança entre os dispositivos e reduzir as vulnerabilidades das aplicações.

8 REFERÊNCIAS

[1] ELKHODARY, A; WHITTLE, J. **A Survey of Approaches to Adaptive Application Security**. In *Proceedings of the 29th international Conference on Software Engineering Workshops* (May 20 - 26, 2007). ICSEW. IEEE Computer Society, Washington, DC, 226.

[2] P. K. MCKINLEY; S. M. SADJADI; E. P. KASTEN; B. H. C. CHENG. **A Taxonomy of Compositional Adaptation.** Technical Report MSU-CSE-04-17, Department of Computer Science and Engineering, Michigan State University, East Lansing, Michigan, 2004.

[3] P. K. MCKINLEY; S. M. SADJADI; E. P. KASTEN; B. H. C. CHENG: **Composing Adaptive Software**" IEEE Computer, 37(7):56- 64, July 2004.

[4] B. HASHII; S. MALABARBA; R. PANDEY; M. BISHOP. **Supporting reconfigurable security policies for mobile programs**. Computer Networks, vol 33(2000), pps. 77-93.

[5] WEISER, M.: **The Computer for the Twenty-First Century**: Scientific American, pp. 94-10, September, 1991.[Ylonen, 2006] Ylonen, T, 2006, RFC 4251 The Secure Shell (SSH) Protocol Architecture.

[6] K. BIRMAN: **The process group approach to reliable distributed computing**. Communications of ACM 36(12), Dec. 1993.

[7] J. KNIGHT; K. SULLIVAN; M. ELDER; C. WANG. **Survivability Architectures: Issues and Approaches.** In DARPA Information Survivability Conference and Exposition. IEEE Computer Society Press. Los Alamitos, CA, January 2000, pp. 157-171.

[8] J. LALA. **Foundations of Intrusion Tolerance Systems**. IEEE Computer Society Press, Catalog # PR02057, 2003.

[9] K. SCOTT, J. DAVIDSON: **Software Security using Software Dynamic Translation**. Technical Report CS-2001-29, Department of Computer Science, University of Virginia, 2001.

[10] P. LAMANNA: **Adaptive Security Policies Enforced by Software Dynamic Translation**. Thesis, University of Virginia. 2002.

[11] K. SCOTT; N. KUMAR, S; VELUSAMY, B; CHILDERS; J. DAVIDSON, M. SOFFA. **Retargetable and Reconfigurable Software Dynamic Translation**. In Code Generation and Optimization (CGO 2003), International Symposium, pps. 36-47, March 2003.

[12] W. HU; J. HISER; D. WILLIAMS; A. FILIPI; J. DAVIDSON; D. EVANS; J. KNIGHT, A. NGUYEN-TUONG, J. ROWANHILL: **Secure and Practical Defense Against Code-injection Attacks using Software Dynamic Translation**.In ACM/Usenix International Conference On Virtual Execution Environments, pps. 2-12, 2006.

[13] KELL, STEPHEN**: A Survey of Practical Software Adaptation Techniques.** Computer Laboratory, University of Cambridge, United Kingdom

[14] J. KNIGHT; D. HEIMBIGNER; A. WOLF;A. CARZANIGA; J. HILL; P. DEVANBU;, M. GERTZ. **The Willow Architecture: Comprehensive Survivability for Large-Scale Distributed Applications.** Intrusion Tolerance Workshop, International Conference on Dependable Systems and Networks, Washington, DC, June 2002

[15] J. KNIGHT, E. STRUNK. **Achieving Critical System Survivability Through Software Architectures.** Workshop on Software Architectures for Dependable Systems, ICSE 2003, pps. 51-78.

[16] D. HEIMBIGNER, A. WOLF. **Intrusion Management Using Configurable Architecture Models.** Department of Computer Science Technical Report CU-CS-929-02 University of Colorado, 2002.

[17] HOH, S., TAN, J.S.; HARTLEY, M. **Context-aware systems - a primer for user-centred services**. BT Technology Journal, Volume 24, Issue 2 (April 2006), pp. 186 – 194, Kluwer Academic Publishers Hingham, MA, USA.

[18] T. RYUTOV; C. NEUMAN. **The Specification and Enforcement of Advanced Security Policies.** 3rd International Workshop on Policies for Distributed Systemsand Networks, pps. 128-138, 2002.

[19] CÁMARA, J; SALAUN, G; CANAL, C. **On Run-time Behavioural Adaptation in Context-Aware Systems.** In: 1ST WORKSHOP ON MODEL-DRIVEN SOFTWARE ADAPTATION (ECOOP 2007). 2007. Berlin, Germany. p. 26-33.

[20] CANAL, C.; MURILLO, J.M.; POIZAT, P. **Software Adaptation**. L'OBJET. Special Issue on Coordination and Adaptation. Techniques for Software Entities. 2006. v. 12, n. 1, p. 9-31.

[21] CARVALHO, W. V.; FERNANDES, P.; TEIXEIRA, R.; ANDRADE, R. M. C. (2005). **Mobile Adapter: Uma abordagem para a construção de Mobile Application Servers adaptativos utilizando as especificações CC/PP e UAProf**. In: Seminário Integrado de Software e Hardware, 32., 2005. São Leopoldo, RS, BRA, p. 1914-1929.

[22] PINHEIRO, MANUELE KIRSCH; VILLANOVA-OLIVER, M.; GENSEL, J.; MARTIN, H. **Awareness on Mobile Groupware Systems**, in the First International Workshop on Mobility Aware Technologies and Applications (MATA 2004), , Florianópolis, Brazil, 20-22 Outubro 2004.

[23] SPRINGER T, KADNER K, STEUER F, YIN M: **Middleware Support for Context-Awareness in 4G.** International Workshop on Wireless Mobile Multimedia archive, Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks, Environments, IEEE Computer Society [Weiser, 1991] Weiser, M.: The Computer for the Twenty-First Century. Scientific American, pp. 94-10, September, 1991.[Ylonen, 2006] Ylonen, T, 2006, RFC 4251 The Secure Shell (SSH) Protocol Architecture.

[24] - BROWN, P.J., BOVEY, J.D. CHEN, X. **Context-Aware Applications**: From the Laboratory to the Marketplace. IEEE Personal Communications, 4(5) (1997) 58-64

[25] J. ZHANG, Z. YANG, B. H. C. CHENG, P. K. MCKINLEY, **Adding safeness to dynamic adaptation techniques**. Proceedings of the ICSE 2004 Workshop on Architecting Dependable Systems, (Edinburgh, Scotland), May 2004.

[26] J. ZHANG, B. H. C. CHENG, Z. YANG, P. K. MCKINLEY. **Enabling safe dynamic component-based software adaptation.** Architecting Dependable Systems III, Springer Lecture Notes for Computer Science (A. R. Rogerio de Lemos, Cristina Gacek, ed.), Springer-Verlag, 2005.

[27] J. ZHANG, BETTY H. C. CHENG, **Model-Based Development of Dynamically Adaptive Software**. IEEE International Conference on Software Engineering (ICSE06), Shanghai, China, May 2006.

[28] – DEY, A.K.; ABOWD, G.D. **Towards a Better Understanding of Context and Context-Awareness.** Atlanta, College of Computing, Geórgia Institute of Technology, 1999. (Relatório Técnico)

[29] - DEY, A.K. **Understanding and Using Context.** ACM Personal and Ubiquitous Computing Journal, v. 5, n.1, p. 4-7, 2001

[30] DIX, A.; RODDEN, T.; DAVIES, N.; TREVOR, J.; FRIDAY, A.; PALFREYMAN, K. **Exploiting Space and Location as a Design Framework for Interactive Mobile Systems.** ACM Transactions on Computer-Human interaction, v.7, n.3, p. 285-321, 2000.

[31] KORTEUM, G.; SEGALL, Z.; BAUER, M. **Context-Aware, Adaptive Wearable Computers.** 2nd International Symposium on Wearable Computers (1998) 58-65

[32] HEATHER, HINTON;, CRISPIN, COWAN; LOIS, DELCAMBRE; SHAWN, BOWERS. **SAM: Security Adaptation Manager**. 15th Annual Computer Security Applications Conference, ACSAC '99, pps 361 – 370, Scottsdale, AZ.

[33] Ambient Intelligence for the Networked Home Environment – Amigo, http://www.hitech-projects.com/euprojects/amigo/software.htm. Acessada em 06/2009.

[34] SCHILIT, B., THEIMER, M. **Disseminating Active Map Information to Mobile Hosts.** IEEE Network, 8(5) (1994) 22-32

[35] SCHILIT, B. N.; ADAMS, N.; WANT, R. **Context-Aware Computing Applications**. In: IEEE WORKSHOP ON MOBILE COMPUTING SYSTEMS AND APPLICATINS, 1., 1994, Santa Cruz. Proceedings. Los Alamitos: IEEE, 1994. p.85-90.

[36] SCHILIT, W.N. **A System Architecture for Context-Aware Mobile Computing,** 1995. 144p. Tese (Doutorado) – Graduate School of Arts and Sciences, Columbia University, Columbia, 1995.

[37] TRUONG, K.N.; ABOWD, G.D.; BROTHERTON, J.A. **Who, What, When, Where, How: Design Issues of Capture & Access Applications.** In: ABOUWD, GD.; BRUMITT,

B.; SHAFER, S. Ubicomp 2001: *International Conference in Ubiquitous Computing*. Heidelberg: Springer-Verlag, 2001. P. 209-224. (Lecture Notes in Computer Science, 2201)
[38] IPINA, D.; KATSIRI, E. **An ECA Rule-Matching Service for Simpler Development of Reactive Applications**. Published as a supplement to the Proc. Of Middleware 2001 at IEEE Distributed Systems Online, Vol. 2, No. 7, November 2001.

[39] WEISE, JOEL Sun Global Systems Engineering Security Office Sun BluePrints™ Online, **Designing an Adaptive Security Architecture**, 2008

[40] DOCKHORN COSTA, P.; PIRES, L. F.; SINDEREN, M., **Architectural Patterns for Context-Aware Services Platforms** in Proceedings of the Second International Workshop on Ubiquitous Computing (IWUC 2005), Miami, May 2005, pp 3-19.

[41] IZQUIERDO, A.; SIERRA, J.M.; TORRES, J. **Providing Security for Digital Ecosystems with Adaptative Encryption.** Digital EcoSystems and Technologies Conference, DEST '07. pps 329 – 333, Australia.

[42] TAYLOR, C.; ALVES-FOSS, J. **Attack Recognition for System Survivability: A Low-level Approach**. Proceedings of the 36th Annual Hawaii International Conference on System Sciences, 2003.

[43] MARCUS, LEO: **Introdution to Logical foundations of an Adaptive Security Infrastucture**, The Aerospace Corporation, Los Angeles, 2004.

[44] BONINO DA SILVA SANTOS, L. O.; RAMPARANY, F.; DOCKHORN COSTA, P.; VINK, P.; ETTER, R.; BROENS, T. H. F. **A Service Architecture for Context Awareness and Reaction Provisioning**. In: 2007 IEEE Congress on Services, 9-13 July 2007, Salt Lake City, UT, USA.